ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ολοκληρωμένο σύστημα χρονοπρογραμματισμού υπεραστικών λεωφορείων και οδηγών

Scheduling System for Intercity Buses and Drivers

Φοιτητής:

Γεώργιος Τζανόπουλος

Α.Μ. 2846

Επιβλέπων Καθηγητής:

Χρήστος Γκόγκος

Καθηγητής

Άρτα, Ιούνιος 2026

Υπεύθυνη Δήλωση

Δηλώνω υπεύθυνα ότι η παρούσα εργασία αποτελεί προϊόν προσωπικής εργασίας και δεν έχει γίνει χρήση πηγών πέραν αυτών που αναφέρονται στη βιβλιογραφία.

Περίληψη

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η ανάπτυξη ενός ολοκληρωμένου υπολογιστικού συστήματος για την υποστήριξη του χρονοπρογραμματισμού υπεραστικών λεωφορείων και οδηγών. Το πρόβλημα προσεγγίζεται μέσω μεταευρετικών αλγορίθμων τοπικής αναζήτησης, με έμφαση στην ταυτόχρονη ανάθεση οδηγών και οχημάτων σε προγραμματισμένα δρομολόγια, λαμβάνοντας υπόψη χρονικούς και εργασιακούς περιορισμούς. Για την εξέταση του προβλήματος υλοποιήθηκαν και συγκρίθηκαν δύο μεταευρετικές τεχνικές: ο αλγόριθμος Late Acceptance Hill Climbing (LAHC) και η Προσομοιωμένη Ανόπτηση (Simulated Annealing). Το σύστημα έχει σχεδιαστεί ώστε να επεξεργάζεται πραγματικά δεδομένα και να παράγει εφικτές λύσεις με στόχο τη μείωση των απαιτούμενων πόρων και του χρόνου αδράνειας, παρέχοντας παράλληλα δυνατότητα απεικόνισης των αποτελεσμάτων μέσω διεπαφής χρήστη.

Λέξεις Κλειδιά: Προγραμματισμός Δρομολογίων, Χρονοπρογραμματισμός Οδηγών, Μεταευρετικοί Αλγόριθμοι, Late Acceptance Hill Climbing, Simulated Annealing, Τοπική Αναζήτηση.

Abstract

This thesis examines the development of an integrated computational system for intercity bus and driver scheduling. The proposed approach uses metaheuristic local search techniques to address the simultaneous assignment of drivers and vehicles to scheduled trips under temporal and labor constraints. Two metaheuristic techniques were implemented and compared: Late Acceptance Hill Climbing (LAHC) and Simulated Annealing. The system processes realworld data and aims to generate feasible solutions while reducing resource usage and idle time, with results presented through a user interface.

Keywords: Route Optimization, Driver Scheduling, Metaheuristic Algorithms, Late Acceptance Hill Climbing, Simulated Annealing, Local Search.

Ευχαριστίες

Θερμές ευχαριστίες εκφράζονται στον κ. Χρήστο Βαλουξή, καθώς η παρούσα εργασία βασίστηκε στο αρχικό του άρθρο και το έργο του σχετικά με τον συνδυασμένο χρονοπρογραμματισμό λεωφορείων και οδηγών.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

1. Εισαγωγή

1.1. Γενικό Υπόβαθρο

Το πρόβλημα του χρονοπρογραμματισμού λεωφορείων και οδηγών (Bus and Driver Scheduling Problem) αποτελεί ένα κλασικό πρόβλημα συνδυαστικής βελτιστοποίησης με σημαντικές πρακτικές εφαρμογές στον τομέα των μεταφορών. Στην Ελλάδα, τα Κοινά Ταμεία Είσπραξης Λεωφορείων (ΚΤΕΛ) διαχειρίζονται καθημερινά εκατοντάδες δρομολόγια υπεραστικών λεωφορείων, αντιμετωπίζοντας την πρόκληση της αποδοτικής κατανομής πόρων ([1]).

Σε αντίθεση με τη διεθνή πρακτική, όπου ο χρονοπρογραμματισμός λεωφορείων και οδηγών αντιμετωπίζεται συχνά ως δύο ξεχωριστά προβλήματα ([2]), στην Ελλάδα παρατηρείται η ιδιαιτερότητα της ενιαίας αντιμετώπισης οδηγού και λεωφορείου ως μίας μονάδας. Κάθε οδηγός είναι συνήθως υπεύθυνος για ένα συγκεκριμένο όχημα, γεγονός που μετατρέπει το πρόβλημα σε μια παραλλαγή του κλασικού Vehicle Routing Problem with Time Windows (VRPTW).

Η μεταβολή των δρομολογίων και οι εργασιακοί περιορισμοί ενισχύουν την ανάγκη για την ανάπτυξη αυτοματοποιημένων συστημάτων. Η χειροκίνητη κατάρτιση προγραμμάτων είναι συχνά χρονοβόρα και μπορεί να οδηγήσει σε υποβέλτιστες λύσεις με αυξημένο λειτουργικό κόστος.

1.2. Κίνητρα και Σημασία του Προβλήματος

Η βελτίωση του χρονοπρογραμματισμού λεωφορείων και οδηγών μπορεί να προσφέρει οφέλη σε πολλαπλά επίπεδα, τα οποία εκτείνονται από την οικονομική βιωσιμότητα της επιχείρησης έως την ασφάλεια.

1.2.1. Οικονομικά Οφέλη

Η εφαρμογή συστημάτων υποστήριξης απόφασης μπορεί να οδηγήσει σε οικονομικά οφέλη μέσω της διαχείρισης του λειτουργικού κόστους. Αυτό μπορεί να επιτευχθεί με τον προγραμματισμό των απαιτούμενων οχημάτων, τον περιορισμό των άσκοπων χιλιομέτρων (deadhead mileage) και τη μείωση των υπερωριών. Επιπλέον, η αυτοματοποίηση του προγραμματισμού μπορεί να συμβάλει στην

αποτελεσματικότερη κατανομή του διοικητικού φόρτου.

1.2.2. Λειτουργικά Οφέλη

Σε επιχειρησιακό επίπεδο, η χρήση υπολογιστικών εργαλείων συμβάλλει στην τήρηση των νομικών και συμβατικών υποχρεώσεων. Η κεντρική διαχείριση των δεδομένων μπορεί να διευκολύνει την ταχεία ανταπόκριση σε αλλαγές, όπως βλάβες οχημάτων ή διακυμάνσεις στη ζήτηση, ενώ παρέχει τη δυνατότητα συλλογής στοιχείων για τη μακροπρόθεσμη αξιολόγηση των υπηρεσιών.

1.2.3. Κοινωνικά Οφέλη

Η εφαρμογή τέτοιων μεθοδολογιών αφορά επίσης την ποιότητα της εργασίας. Η δικαιότερη κατανομή του φόρτου εργασίας μπορεί να συμβάλει στην ικανοποίηση του προσωπικού και στη μείωση της κόπωσης, γεγονός που σχετίζεται με την οδική ασφάλεια. Για το επιβατικό κοινό, η συνέπεια στα δρομολόγια μπορεί να ενισχύσει την αξιοπιστία των δημόσιων συγκοινωνιών.

1.3. Συνεισφορά της Εργασίας

Η παρούσα εργασία εξετάζει το πρόβλημα του ταυτόχρονου χρονοπρογραμματισμού οχημάτων και οδηγών (Integrated Vehicle and Crew Scheduling Problem) με έμφαση στις ιδιαιτερότητες των ελληνικών υπεραστικών συγκοινωνιών (ΚΤΕΛ).

Σε αυτό το πλαίσιο, εξετάζεται η αποτελεσματικότητα δύο μεταευρετικών αλγορίθμων, του Simulated Annealing (SA) και του Late Acceptance Hill Climbing (LAHC), σε σενάρια δρομολογίων. Επιπλέον, αναπτύχθηκε πρωτότυπο πληροφοριακού συστήματος που περιλαμβάνει έναν solver σε Python, ένα API για τη διαχείριση των αιτημάτων και μια διεπαφή χρήστη για την απεικόνιση των βαρδιών. Το σύστημα επιδιώκει να υποστηρίζει σύνθετους περιορισμούς, όπως η συνεχής ροή σταθμών (station chaining, δηλαδή η απαίτηση το επόμενο δρομολόγιο ενός οδηγού/λεωφορείου να ξεκινά υποχρεωτικά από τον σταθμό στον οποίο τερμάτισε το αμέσως προηγούμενο), τα διαλείμματα και τα όρια υπερωριών, καθιστώντας το εργαλείο πρακτικά εφαρμόσιμο από έναν οργανισμό συγκοινωνιών.

2. Θεωρητικό Υπόβαθρο

2.1. Εισαγωγή

Το πρόβλημα του χρονοπρογραμματισμού λεωφορείων και οδηγών (Bus and Driver Scheduling Problem - BDSP) αποτελεί ένα σύνθετο πρόβλημα βελτιστοποίησης που συνδυάζει δύο κλασικά πεδία της Επιχειρησιακής Έρευνας: τη Δρομολόγηση Οχημάτων (Vehicle Routing) και τον Χρονοπρογραμματισμό Πληρωμάτων (Crew Scheduling). Σε αυτό το κεφάλαιο παρουσιάζονται οι βασικές έννοιες και η βιβλιογραφική ανασκόπηση που θεμελιώνουν την προσέγγιση της παρούσας εργασίας.

2.2. Βασικές Έννοιες

2.2.1. Προβλήματα χρονοπρογραμματισμού

Τα προβλήματα χρονοπρογραμματισμού είναι μια ευρεία κατηγορία προβλημάτων συνδυαστικής βελτιστοποίησης, στα οποία ζητείται η χρονική τοποθέτηση εργασιών, γεγονότων ή ανθρώπινων πόρων σε διαθέσιμους πόρους, υπό περιορισμούς διαθεσιμότητας, χωρητικότητας, αλληλουχίας και κόστους. Ο όρος scheduling χρησιμοποιείται συνήθως για την ανάθεση και διάταξη εργασιών σε μηχανές, επεξεργαστές ή παραγωγικούς πόρους, όπως σε προβλήματα χρονοπρογραμματισμού εργασιών σε επεξεργαστές και σε προβλήματα τύπου permutation flow-shop [3, 4]. Ο όρος timetabling αφορά κυρίως τη δημιουργία ωρολογίων προγραμμάτων με προκαθορισμένες χρονικές θέσεις και συγκρούσεις πόρων, όπως στην κατάρτιση προγραμμάτων εξετάσεων ή σχολικών προγραμμάτων μαθημάτων [5, 6], ενώ το rostering εστιάζει στην ανάθεση βαρδιών σε προσωπικό σε μεγαλύτερο χρονικό ορίζοντα, με χαρακτηριστικό παράδειγμα το nurse rostering [7]. Παρά τις επιμέρους διαφορές τους, τα παραπάνω προβλήματα έχουν κοινή δομή, καθώς απαιτούν την αντιστοίχιση δραστηριοτήτων σε χρονικές θέσεις και πόρους, την ικανοποίηση αυστηρών περιορισμών και τη βελτιστοποίηση κριτηρίων ποιότητας, όπως το κόστος, η ισορροπία φόρτου εργασίας και οι υπερωρίες. Στον χώρο των μεταφορών, η σύνδεση με τα προβλήματα δρομολόγησης οχημάτων είναι άμεση, διότι κάθε δρομολόγιο δεν αποτελεί μόνο μετακίνηση στο χώρο αλλά

και μια εργασία που είναι χρονικά δεσμευμένη. Έτσι, η επιλογή της σειράς των διαδρομών, των χρόνων αναχώρησης, των ενδιάμεσων στάσεων και της ανάθεσης οδηγών και οχημάτων μετατρέπει τα προβλήματα δρομολόγησης οχημάτων και τις παραλλαγές τους με παράθυρα χρόνου σε προβλήματα χρονοπρογραμματισμού με ενσωματωμένη τη διάσταση του χώρου.

2.2.2. Δρομολόγηση Οχημάτων (Vehicle Routing Problem - VRP)

Το VRP ([8]) αφορά την εύρεση βέλτιστων διαδρομών για έναν στόλο οχημάτων που πρέπει να εξυπηρετήσουν ένα σύνολο πελατών. Στην περίπτωση των λεωφορείων, το πρόβλημα εστιάζει στη χρονική και χωρική αλληλουχία των δρομολογίων, διασφαλίζοντας ότι κάθε λεωφορείο βρίσκεται στη σωστή αφετηρία τη σωστή στιγμή, τηρώντας παράλληλα περιορισμούς χωρητικότητας και χρονικών παραθύρων (VRPTW).

2.2.3. Χρονοπρογραμματισμός Πληρωμάτων (Crew Scheduling Problem - CSP)

Το CSP ([9]) εστιάζει στην ανάθεση εργασιών σε ανθρώπινο δυναμικό. Η πολυπλοκότητά του πηγάζει από τους αυστηρούς εργασιακούς κανονισμούς (μέγιστες ώρες οδήγησης, υποχρεωτικά διαλείμματα, ελάχιστη ανάπαυση). Παραδοσιακά, το πρόβλημα επιλύεται σε δύο στάδια: τη δημιουργία βαρδιών (Pairing) και την ανάθεση αυτών σε συγκεκριμένους οδηγούς (Rostering).

2.3. Η Ιδιαιτερότητα της Ελληνικής Περίπτωσης

Η προσέγγιση της παρούσας εργασίας βασίζεται στην εργασία των Valouxis και Housos (2002) ([1]), που αναλύει το πρόβλημα στα ελληνικά ΚΤΕΛ. Η κύρια διαφορά από τη διεθνή βιβλιογραφία είναι η ενιαία αντιμετώπιση οδηγού και λεωφορείου.

Στην Ελλάδα, ο οδηγός είναι συνήθως ιδιοκτήτης ή αποκλειστικός χρήστης του λεωφορείου του. Αυτό συνεπάγεται τις εξής ιδιαιτερότητες:

1. **Συνδρομή Πόρων:** Το όχημα και ο οδηγός δεν χωρίζονται κατά τη διάρκεια της βάρδιας, αποτελώντας μια ενιαία λειτουργική μονάδα.
2. **Ενοποίηση Προβλημάτων:** Το πρόβλημα μετατρέπεται από δύο ανεξάρτητα προβλήματα (CSP/VRP) σε ένα ενοποιημένο πρόβλημα ανάθεσης δρομολογίων

σε σταθερά ζεύγη οδηγού-οχήματος.

3. **Περιορισμοί Συνοχής:** Οι περιορισμοί «χρονικής και χωρικής συνοχής» γίνονται κρίσιμοι και ορίζονται ως εξής:

    - **Χωρική συνοχή (spatial continuity):** Απαιτεί η αφετηρία κάθε δρομολογίου να συμπίπτει με την τοποθεσία στην οποία βρίσκεται το όχημα (δηλαδή τον σταθμό άφιξης του προηγούμενου δρομολογίου ή τον σταθμό στον οποίο έχει μεταβεί μέσω νεκρής διαδρομής).

    - **Χρονική συνοχή (temporal continuity):** Απαιτεί η ώρα έναρξης κάθε δρομολογίου να είναι μεταγενέστερη από την ώρα άφιξης του προηγούμενου, συνυπολογίζοντας τον χρόνο που απαιτείται για τη μετάβαση μεταξύ σταθμών και τα υποχρεωτικά διαλείμματα.

2.4. Μεθοδολογίες Επίλυσης

Λόγω της NP-πληρότητας του προβλήματος ([10, 11]), η βιβλιογραφία προτείνει δύο κύριες κατηγορίες μεθόδων:

2.4.1. Ακριβείς Μέθοδοι (Exact Methods)

Μέθοδοι όπως το Column Generation ([1]) στοχεύουν στην εύρεση της μαθηματικά βέλτιστης λύσης. Αν και παρέχουν εγγυήσεις ποιότητας, συχνά απαιτούν υπερβολικό υπολογιστικό χρόνο για μεγάλα δίκτυα δρομολογίων, καθιστώντας τις δύσχρηστες για καθημερινή επιχειρησιακή χρήση.

2.4.2. Μεταευρετικοί Αλγόριθμοι (Metaheuristics)

Οι μεταευρετικοί αλγόριθμοι αποτελούν μια συνηθισμένη επιλογή για τέτοιου είδους προβλήματα λόγω της ταχύτητάς τους και της δυνατότητας διαχείρισης σύνθετων περιορισμών.

- Προσομοιωμένη Ανόπτηση (Simulated Annealing): Μια στοχαστική μέθοδος που επιτρέπει προσωρινά την αποδοχή χειρότερων λύσεων με στόχο την αποφυγή τοπικών ακροτάτων.

- Late Acceptance Hill Climbing (LAHC): Μια παραλλαγή τοπικής αναζήτησης που χρησιμοποιεί ιστορικό προηγούμενων λύσεων για την απόφαση αποδοχής νέων κινήσεων.

2.4.3. Προσεγγίσεις Βασισμένες σε Γράφους και Προβλήματα Ανάθεσης

Μια εναλλακτική προσέγγιση για την επίλυση του BDSP εστιάζει στη μοντελοποίηση του προβλήματος με χρήση γράφων και την επίλυσή του μέσω διαδοχικών προβλημάτων γραμμικής ανάθεσης (linear assignment problems). Ο Constantino και οι συνεργάτες του (2017) ([12]) πρότειναν τον ντετερμινιστικό αλγόριθμο δύο φάσεων GraphBDSP, ο οποίος βασίζεται σε έναν προσανατολισμένο πολυμερή γράφο (multipartite directed graph). Στο μοντέλο αυτό, οι προς ανάθεση εργασίες αναπαρίστανται ως κορυφές κατανεμημένες σε επίπεδα (layers), ενώ οι ακμές εκφράζουν τις εφικτές μεταβάσεις μεταξύ των εργασιών.

Στην πρώτη φάση (constructive phase), ο αλγόριθμος κατασκευάζει μια αρχική εφικτή λύση επιλύοντας διαδοχικά προβλήματα ανάθεσης σε πολυωνυμικό χρόνο. Στη δεύτερη φάση (improvement phase), η λύση βελτιώνεται μέσω δύο τοπικών διαδικασιών αναζήτησης: η πρώτη (Imp1) διαιρεί τις βάρδιες σε επιμέρους τμήματα και τις επανασυνδυάζει επιλύοντας εκ νέου προβλήματα ανάθεσης, ενώ η δεύτερη (Imp2) στοχεύει στη μείωση των υπερωριών μεταφέροντας εργασίες από επιβαρυμένες βάρδιες. Η μελέτη αυτή ανέλυσε τη συμπεριφορά διαφορετικών αντικειμενικών συναρτήσεων κόστος, καταλήγοντας στο συμπέρασμα ότι η εστίαση στον χρόνο αδράνειας και τις υπερωρίες οδηγεί σε ποιοτικότερα αποτελέσματα σε σύγκριση με την αποκλειστική ελαχιστοποίηση του καθαρού πληρωτέου χρόνου. Το σημαντικότερο στοιχείο της συγκεκριμένης εργασίας είναι η επιτυχής εφαρμογή της σε πραγματικά δεδομένα μεγάλης κλίμακας (άνω των 2.300 εργασιών), αναδεικνύοντας τη σημασία των ευρετικών μεθόδων έναντι των ακριβών προσεγγίσεων, οι οποίες συχνά παρουσιάζουν περιορισμούς λόγω της υπολογιστικής πολυπλοκότητας του προβλήματος.

2.5. Σύνοψη

Η προσέγγιση του BDSP απαιτεί τη σύνθεση των κανόνων δρομολόγησης και των εργασιακών περιορισμών. Στην ελληνική πραγματικότητα, η ενιαία διαχείριση οδηγού-λεωφορείου επηρεάζει τη δομή της λύσης. Σε αυτή την εργασία υιοθετείται μια μεταευρετική προσέγγιση για την παροχή αποδοτικών λύσεων σε αυτό το πλαίσιο.

3. Μαθηματική Διατύπωση του Προβλήματος

3.1. Εισαγωγή

Σε αυτό το κεφάλαιο παρουσιάζεται η επίσημη μαθηματική διατύπωση του προβλήματος χρονοπρογραμματισμού λεωφορείων και οδηγών για τα ελληνικά ΚΤΕΛ. Ορίζονται τα σύνολα, οι παράμετροι, οι μεταβλητές απόφασης, οι περιορισμοί και η αντικειμενική συνάρτηση, και αποδεικνύεται η NP-πληρότητα του προβλήματος.

3.2. Σύνολα και Δείκτες

Για τη μαθηματική αναπαράσταση του προβλήματος, ορίζουμε το σύνολο $T = \{1, 2, \dots, n\}$ που περιλαμβάνει όλα τα δρομολόγια (trips) προς εκτέλεση και το σύνολο $B = \{1, 2, \dots, m\}$ για τα διαθέσιμα λεωφορεία. Οι οδηγοί συμβολίζονται με το σύνολο $D = \{1, 2, \dots, m\}$, όπου λόγω της 1-1 αντιστοίχισης ισχύει $|D| = |B|$. Το σύνολο των σταθμών (stations) ορίζεται ως $S = \{s_1, s_2, \dots, s_k\}$. Επιπλέον, το σύνολο $\mathcal{P}$ περιλαμβάνει τις υποχρεωτικές ακολουθίες δρομολογίων (mandatory sequences), ενώ το σύνολο $\mathcal{F} \subseteq T \times B$ περιλαμβάνει τις απαγορευμένες αναθέσεις μεταξύ δρομολογίων και λεωφορείων (για παράδειγμα, λόγω γεωγραφικών περιορισμών όπως στενοί δρόμοι ή χαμηλές γέφυρες που απαγορεύουν τη διέλευση μεγάλων οχημάτων, ή λόγω έλλειψης ειδικού εξοπλισμού όπως πρόσβαση ΑμεΑ σε συγκεκριμένες γραμμές).

3.3. Παράμετροι Δεδομένων

3.3.1. Παράμετροι Δρομολογίων

Για κάθε δρομολόγιο $i \in T$, ορίζονται οι εξής παράμετροι: ο χρόνος εκκίνησης $s_i \in \mathbb{Z}^+$ και ο χρόνος λήξης $e_i \in \mathbb{Z}^+$ σε λεπτά από τα μεσάνυχτα ($0 \leq s_i < 1440$), η διάρκεια οδήγησης $d_i = e_i - s_i$, ο σταθμός εκκίνησης $o_i \in S$ και ο σταθμός τερματισμού $t_i \in S$, καθώς και ο απαιτούμενος τύπος λεωφορείου $type_i \in \{01, 02, \dots, 07\}$ (όπου οι κωδικοί αυτοί αντιστοιχούν σε διαφορετικές κατηγορίες οχημάτων, π.χ., κανονικά λεωφορεία, mini-bus κ.λπ. βάσει των δεδομένων των ΚΤΕΛ).

3.3.2. Παράμετροι Λεωφορείων/Οδηγών

Για κάθε λεωφορείο/οδηγό $j \in B$, ορίζεται η έδρα (home station) $home_j \in S$, ο τύπος οχήματος $type_j$ και η νωρίτερη δυνατή ώρα έναρξης βάρδιας $shift\_start_j \in \mathbb{Z}^+$.

3.3.3. Γενικές Παράμετροι Συστήματος

Οι γενικές παράμετροι του συστήματος ορίζονται στην κλάση `SolverParameters` (στο αρχείο legality_checker.py) και καθορίζουν τα χρονικά όρια των βαρδιών. Οι τιμές αυτές συνοψίζονται στον Πίνακα 3.1.

Πίνακας 3.1: Γενικές Παράμετροι Συστήματος

Σύμβολο	Περιγραφή
D_{max}	Μέγιστη διάρκεια βάρδιας
D_{normal}	Κανονική διάρκεια βάρδιας
W_{max}	Μέγιστος χρόνος εργασίας
B_{min}	Ελάχιστο διάλειμμα
τ	Χρόνος μετάβασης (deadhead) / Ελάχιστος χρόνος αναμονής

3.3.4. Βάρη Αντικειμενικής Συνάρτησης

Για τη στάθμιση των διαφορετικών στόχων της αντικειμενικής συνάρτησης χρησιμοποιούνται τα βάρη που ορίζονται στην κλάση `SAWeights` (στο αρχείο simulated_annealing.py). Η επιλογή των τιμών αυτών καθορίζει την προτεραιότητα του αλγορίθμου κατά τη βελτιστοποίηση, όπως φαίνεται στον Πίνακα 3.2.

Πίνακας 3.2: Βάρη Αντικειμενικής Συνάρτησης (SAWeights)

Σύμβολο	Στόχος Βελτιστοποίησης	Βάρος (Ποινή)
$w_{unassigned}$	Μη ανατεθειμένα δρομολόγια	100,000,000.0
w_{bus}	Χρήση επιπλέον λεωφορείων	50,000.0
$w_{overtime}$	Υπερωρία (ανά λεπτό > 10 ώρες)	100.0
w_{return}	Ποινή μη επιστροφής στη βάση	2,000.0
$w_{fairness}$	Δίκαιη κατανομή φόρτου (fairness)	5.0
w_{idle}	Νεκρός χρόνος (ανά λεπτό)	1.0

3.4. Μεταβλητές Απόφασης

Ορίζουμε τις εξής δυαδικές μεταβλητές απόφασης:

$$x_{ij} = \begin{cases} 1 & \text{αν το δρομολόγιο } i \text{ ανατεθεί στο λεωφορείο/οδηγό } j \\ 0 & \text{διαφορετικά} \end{cases} \quad \forall i \in T, j \in B$$

3.4.1. Παράγωγες Μεταβλητές

Για διευκόλυνση της διατύπωσης, ορίζουμε: Σύνολο δρομολογίων ανά λεωφορείο:

$$T_j = \{i \in T : x_{ij} = 1\} \quad \forall j \in B$$

Διάρκεια βάρδιας (duty duration) λεωφορείου j:

$$duty_j = \begin{cases} e_{last(j)} - s_{first(j)} & \text{αν } |T_j| > 0 \\ 0 & \text{διαφορετικά} \end{cases}$$

όπου $first(j) = \arg\min_{i \in T_j} s_i$ και $last(j) = \arg\max_{i \in T_j} e_i$. Σημείωση: Αν $e_{last(j)} < s_{first(j)}$ (δηλ. η βάρδια περνά τα μεσάνυχτα), τότε $duty_j = e_{last(j)} + 1440 - s_{first(j)}$.

Νεκρός χρόνος (idle time) λεωφορείου j:

$$idle_j = \sum_{\substack{i,i' \in T_j \\ s_{i'} > s_i}} \max(0, s_{i'} - e_i - \tau \cdot \mathbf{1}[t_i \neq o_{i'}])$$

όπου $\mathbf{1}[t_i \neq o_{i'}] = 1$ αν το τέλος του i δεν συμπίπτει με την αρχή του i'.

Υπερωρίες πέρα από 10 ώρες:

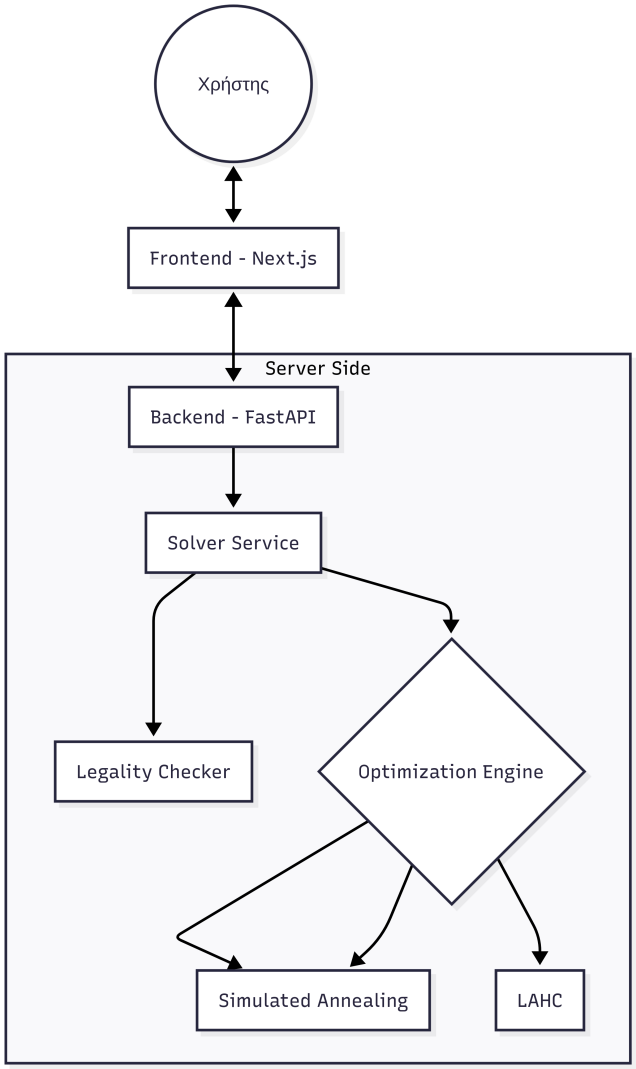
$$overtime_j = \max(0, duty_j - D_{normal})$$

Δείκτης χρήσης λεωφορείου:

$$used_j = \begin{cases} 1 & \text{αν } |T_j| > 0 \\ 0 & \text{διαφορετικά} \end{cases}$$

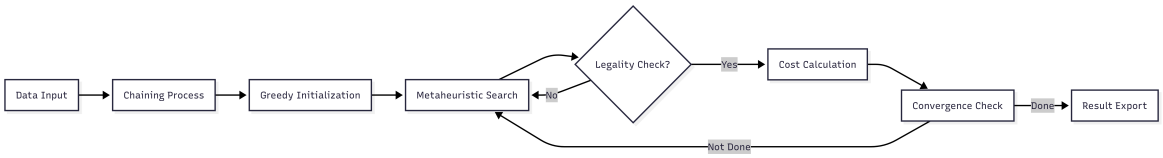
3.5. Αρχιτεκτονική και Ροή Δεδομένων

Η αρχιτεκτονική του συστήματος δίνει προτεραιότητα στην επεκτασιμότητα και την άμεση ανταπόκριση, στοιχεία απαραίτητα για ένα επιχειρησιακό εργαλείο. Στα σχήματα που ακολουθούν αποτυπώνεται η δομή και η εσωτερική ροή λειτουργίας του επιλύτη.



Σχήμα 3.1: Αρχιτεκτονική Συστήματος: Διαχωρισμός Backend (FastAPI) και Frontend (Next.js).

Η ροή των δεδομένων ξεκινά από το Frontend, όπου ο χρήστης εισάγει ή τροποποιεί τα δρομολόγια. Το αρχείο JSON αποστέλλεται στο Backend, όπου το Pydantic αναλαμβάνει την επικύρωση των τύπων δεδομένων. Στη συνέχεια, ο Solver εκτελεί τον επιλεγμένο αλγόριθμο (SA ή LAHC) και επιστρέφει ένα δομημένο αντικείμενο με τις αναθέσεις και τις μετρικές απόδοσης.



Σχήμα 3.2: Ροή Λειτουργίας Επιλύτη: Από την ανάγνωση δεδομένων έως την εξαγωγή βέλτιστης λύσης.

3.6. Αντικειμενική Συνάρτηση

Κεντρικός στόχος του συστήματος είναι η ελαχιστοποίηση της συνάρτησης συνολικού κόστους (Z). Το κόστος αυτό δεν αντιστοιχεί σε χρηματικές μονάδες, αλλά λειτουργεί ως ένας μηχανισμός επιβολής ποινών που καθοδηγεί τη διαδικασία βελτιστοποίησης προς λύσεις που ικανοποιούν τις επιχειρησιακές προτεραιότητες.

3.6.1. Ανάλυση και Στρατηγική Επιλογής Βαρών

Η επιλογή των βαρών (w_k) στην αντικειμενική συνάρτηση (Πίνακας 3.2) δεν είναι τυχαία, αλλά αντικατοπτρίζει μια αυστηρή ιεράρχηση των επιχειρησιακών στόχων. Η φιλοσοφία βασίζεται στην έννοια των «επιπέδων ποινής»:

- **Επίπεδο 1 - Απόλυτη Εφικτότητα ($w_{\text{unassigned}} = 10^8$):** Η κάλυψη όλων των δρομολογίων είναι ο υπέρτατος στόχος. Μια λύση που αφήνει έστω και ένα δρομολόγιο ακάλυπτο θεωρείται πρακτικά άχρηστη, επομένως η ποινή είναι τόσο υψηλή που ο αλγόριθμος την αποφεύγει πάση θυσία.
- **Επίπεδο 2 - Οικονομία Πόρων ($w_{\text{bus}} = 50.000$):** Η μείωση του αριθμού των λεωφορείων και οδηγών είναι η κύρια πηγή οικονομικού οφέλους. Η ποινή αυτή είναι αρκετά μεγάλη ώστε να ωθεί τον solver να «στριμώχνει» δρομολόγια σε λιγότερες βάρδιες, αλλά υποχωρεί μπροστά στον κίνδυνο να μείνει δρομολόγιο ακάλυπτο.
- **Επίπεδο 3 - Λειτουργική Ποιότητα ($w_{\text{return}} = 2.000$, $w_{\text{overtime}} = 100$):** Εδώ

εντάσσονται οι περιορισμοί που κάνουν το πρόγραμμα «βιώσιμο». Η αποφυγή διανυκτέρευσης εκτός έδρας (μέσω της ποινής μη επιστροφής στη βάση) έχει προτεραιότητα έναντι των υπερωριών, καθώς η διανυκτέρευση εκτός έδρας συνεπάγεται υψηλά κόστη διαμονής και οργανωτική αναστάτωση.

- **Επίπεδο 4 - Βελτιστοποίηση δίκαιης κατανομής εργασίας ($w_{\text{fairness}} = 5$, $w_{\text{idle}} = 1$):** Η δίκαιη κατανομή (fairness) στοχεύει στην ισομερή κατανομή του φόρτου εργασίας μεταξύ των οδηγών, εξασφαλίζοντας ότι οι διάρκειες των βαρδιών προσεγγίζουν έναν κοινό μέσο όρο και αποφεύγονται οι ακραίες αποκλίσεις (δηλαδή η ανάθεση υπερβολικά μεγάλων βαρδιών σε ορισμένους οδηγούς έναντι πολύ μικρών σε άλλους). Αυτό, σε συνδυασμό με την ελαχιστοποίηση του νεκρού χρόνου, αποτελεί το τελικό στάδιο ραφιναρίσματος της λύσης για τη βελτίωση των συνθηκών εργασίας του προσωπικού.

Η αντικειμενική συνάρτηση διαμορφώνεται ως το σταθμισμένο άθροισμα των παρακάτω συνιστωσών:

1. **Κόστος Ανεκχώρητων Δρομολογίων (Z_u):** Αντιπροσωπεύει το πλήθος των δρομολογίων που έμειναν χωρίς ανάθεση:

$$Z_u = \sum_{i \in T} \left(1 - \sum_{j \in B} x_{ij}\right)$$

2. **Κόστος Χρησιμοποιούμενων Λεωφορείων/Οδηγών (Z_d):** Αντιπροσωπεύει το συνολικό πλήθος των οδηγών/λεωφορείων που χρησιμοποιήθηκαν στο πρόγραμμα:

$$Z_d = \sum_{j \in B} used_j$$

3. **Κόστος Υπερωριών (Z_o):** Επιβάλλει ποινή για τον συνολικό χρόνο υπερωριακής απασχόλησης (σε λεπτά) πέραν της κανονικής διάρκειας βάρδιας:

$$Z_o = \sum_{j \in B} overtime_j$$

4. **Κόστος Νεκρού Χρόνου (Z_i):** Μετρά τον συνολικό χρόνο αδράνειας (σε

λεπτά) των οδηγών/λεωφορείων κατά τη διάρκεια των βαρδιών τους:

$$Z_i = \sum_{j \in B} idle_j$$

5. **Κόστος Δικαιοσύνης (Fairness - Z_f):** Χρησιμοποιεί μια τετραγωνική ποινή απόκλισης της διάρκειας κάθε βάρδιας από τη μέση επιθυμητή διάρκεια βάρδιας $\bar{d}$ για την εξισορρόπηση του φόρτου εργασίας:

$$Z_f = \sum_{j \in B} (d_j - \bar{d})^2$$

όπου d_j η διάρκεια βάρδιας του οδηγού j και $\bar{d}$ η μέση επιθυμητή διάρκεια.

6. **Ποινή Μη Επιστροφής στη Βάση (Z_r):** Αντιπροσωπεύει το πλήθος των οδηγών/λεωφορείων που δεν ολοκληρώνουν τη βάρδιά τους στην έδρα τους (base mismatch):

$$Z_r = \sum_{j \in B} \mathbf{1}[t_{last(j)} \neq home_j]$$

Η τελική μορφή της αντικειμενικής συνάρτησης είναι η ελαχιστοποίηση του συνολικού σταθμισμένου κόστους:

$$\text{min } Z = w_u Z_u + w_d Z_d + w_o Z_o + w_r Z_r + w_f Z_f + w_i Z_i$$

όπου οι συντελεστές στάθμισης $w_u, w_d, w_o, w_r, w_f, w_i$ αντιστοιχούν στα βάρη του Πίνακα 3.2 ($w_{unassigned}, w_{bus}, w_{overtime}, w_{return}, w_{fairness}, w_{idle}$ αντίστοιχα).

3.7. Περιορισμοί

Το πρόβλημα που εξετάζουμε διέπεται από ένα σύνολο περιορισμών, οι οποίοι χωρίζονται σε δύο κύριες κατηγορίες: τους σκληρούς (Hard Constraints) και τους χαλαρούς (Soft Constraints).

3.7.1. Hard Constraints

Οι Hard Constraints αποτελούν τις απαραίτητες προϋποθέσεις που πρέπει να ικανοποιούνται πλήρως ώστε μια λύση να θεωρείται έγκυρη και εφαρμόσιμη στην πράξη. Οποιαδήποτε παραβίαση αυτών των κανόνων καθιστά το πρόγραμμα δρομολογίων άκυρο. Οι βασικότεροι περιορισμοί αυτής της κατηγορίας περιγράφονται παρακάτω.

3.7.1.1 C1: Κάλυψη Όλων των Δρομολογίων

$$\sum_{j=1}^{m} x_{ij} = 1, \quad \forall i \in T$$

Κάθε δρομολόγιο πρέπει να ανατεθεί ακριβώς σε ένα λεωφορείο.

3.7.1.2 C2: Αποκλειστικότητα (Μη Επικάλυψη)

Για κάθε λεωφορείο j και κάθε ζεύγος δρομολογίων $i, i' \in T_j$ με $i < i'$:

$$[s_i, e_i) \cap [s_{i'}, e_{i'}) = \emptyset$$

Δηλαδή, τα χρονικά διαστήματα των δρομολογίων δεν πρέπει να επικαλύπτονται.

3.7.1.3 C3: Χρονική Συνοχή

Για κάθε $j \in B$ και για κάθε δύο διαδοχικά δρομολόγια i, i' στο T_j (ταξινομημένα ως προς s):

$$s_{i'} \geq e_i + \tau \cdot \mathbf{1}[t_i \neq o_{i'}]$$

Το επόμενο δρομολόγιο πρέπει να ξεκινά μετά το τέλος του προηγούμενου, προσθέτοντας χρόνο μετάβασης αν οι σταθμοί διαφέρουν.

3.7.1.4 C4: Χωρική Συνοχή

Για κάθε $j \in B$ και για κάθε δύο διαδοχικά δρομολόγια $i, i' \in T_j$:

$$t_i = o_{i'} \quad \text{και} \quad s_{i'} \geq e_i + \tau$$

Η χωρική συνοχή επιβάλλει ότι το επόμενο δρομολόγιο i' πρέπει να ξεκινά από την ίδια τοποθεσία στην οποία τερμάτισε το προηγούμενο δρομολόγιο i ($t_i = o_{i'}$), ενώ παράλληλα απαιτείται ένας ελάχιστος χρόνος αναμονής/προετοιμασίας τ μεταξύ τους ($s_{i'} \geq e_i + \tau$).

Σημειώνεται ότι, στην υλοποίηση του συστήματος (αρχείο `legality_checker.py`), οι ενδιάμεσες μεταβάσεις (deadheads) μεταξύ διαφορετικών σταθμών κατά τη διάρκεια της ίδιας βάρδιας απαγορεύονται αυστηρά (μέσω του ελέγχου `station_mismatch`). Ο όρος **deadhead** (νεκρή μετάβαση) αναφέρεται στην κενή μετακίνηση του λεωφορείου χωρίς επιβάτες, η οποία στο συγκεκριμένο μοντέλο επιτρέπεται μόνο κατά την έναρξη της βάρδιας (από τη βάση προς το πρώτο δρομολόγιο, ελεγχόμενη από τον πίνακα `stationTransferMatrix`) και κατά τη λήξη της (επιστροφή στη βάση, η οποία ελέγχεται και ποινικοποιείται στην αντικειμενική συνάρτηση μέσω του όρου Z_r).

3.7.1.5 C5: Συμβατότητα Τύπου Λεωφορείου

$$x_{ij} = 1 \Rightarrow type_j \geq type_i, \quad \forall i \in T, j \in B$$

Το λεωφορείο πρέπει να είναι τουλάχιστον του ίδιου τύπου (μεγέθους) με αυτό που απαιτεί το δρομολόγιο.

3.7.1.6 C6: Μέγιστη Διάρκεια Βάρδιας

$$duty_j \leq D_{max}, \quad \forall j \in B$$

Η συνολική διάρκεια βάρδιας δεν μπορεί να υπερβαίνει τα $D_{max} = 900$ λεπτά (15 ώρες).

3.7.1.7 C7: Νωρίτερη Ώρα Έναρξης

$$s_{first(j)} \geq shift\_start_j, \quad \forall j \in B \text{ με } |T_j| > 0$$

Το πρώτο δρομολόγιο δεν μπορεί να ξεκινά πριν τη νωρίτερη επιτρεπόμενη ώρα του οδηγού.

3.7.1.8 C8: Υποχρεωτικές Ακολουθίες

Για κάθε ακολουθία $\mathcal{P}_p = \{i_1, i_2, ..., i_k\} \in \mathcal{P}$:

$$x_{i_1 j} = x_{i_2 j} = ... = x_{i_k j}, \quad \exists j \in B$$

Όλα τα δρομολόγια της ακολουθίας πρέπει να ανατεθούν στο ίδιο λεωφορείο.

3.7.1.9 C9: Απαγορευμένες Αναθέσεις

$$x_{ij} = 0, \quad \forall (i, j) \in \mathcal{F}$$

Ορισμένες αναθέσεις μπορεί να απαγορεύονται (π.χ. λόγω ασυμβατότητας οδηγού με δρομολόγιο).

Σημείωση Υλοποίησης: Οι παραπάνω θεωρητικοί περιορισμοί (C1–C9) ενσωματώνονται στο τελικό σύστημα μέσω 15 εξειδικευμένων ελέγχων (K1–K15) στον Legality Checker, διασφαλίζοντας την τήρηση τόσο των μαθηματικών όσο και των εργατικών/επιχειρησιακών κανόνων.

3.7.2. Χαλαροί Περιορισμοί (Soft Constraints)

Σε αντίθεση με τους σκληρούς περιορισμούς, οι χαλαροί (Soft Constraints) περιγράφουν ιδιότητες που είναι επιθυμητές αλλά όχι υποχρεωτικές για τη λειτουργία του συστήματος. Η παραβίαση αυτών των κανόνων δεν ακυρώνει τη λύση, αλλά επιφέρει μια ”ποινή” στην αντικειμενική συνάρτηση, την οποία ο αλγόριθμος προσπαθεί να ελαχιστοποιήσει.

3.7.2.1 S1: Τερματισμός στην Έδρα

Είναι επιθυμητό:

$$t_{last(j)} = home_j, \quad \forall j \in B$$

Αν δεν ικανοποιείται, προστίθεται ποινή στο κόστος (μπορεί να ενσωματωθεί στο Z_{idle}).

3.7.2.2 S2: Κανονική Διάρκεια Βάρδιας

Είναι επιθυμητό:

$$duty_j \approx D_{normal} = 600 \text{ λεπτά}$$

Αποκλίσεις τιμωρούνται μέσω του Z_{fair} και $Z_{overtime}$.

3.7.2.3 S3: Ελαχιστοποίηση Νεκρού Χρόνου

Ελαχιστοποιείται άμεσα μέσω του Z_{idle}.

3.7.2.4 S4: Επιθυμίες Οδηγών (Driver Wishes)

Ορισμένοι οδηγοί μπορεί να έχουν εκφράσει επιθυμίες για συγκεκριμένα δρομολόγια ή βάρδιες. Αυτό μοντελοποιείται μέσω θετικής επιθυμίας (bonus στο κόστος αν $x_{ij} = 1$) ή αρνητικής επιθυμίας (ποινή στο κόστος αν $x_{ij} = 1$). Στον κώδικα, η λειτουργία αυτή υλοποιείται στη συνάρτηση `check_driver_wish_bonus()` (βλ. `legality_checker.py`).

3.8. Πολυπλοκότητα του Προβλήματος

3.8.1. Θεώρημα: NP-πληρότητα

Θεώρημα 1: Το Πρόβλημα Χρονοπρογραμματισμού Λεωφορείων και Οδηγών (BDSP) είναι NP-πλήρες.

Απόδειξη: Θα δείξουμε ότι το BDSP γενικεύει το Vehicle Routing Problem with Time Windows (VRPTW), το οποίο είναι γνωστό ότι είναι NP-πλήρες ([13]). Αναγωγή από VRPTW σε BDSP: Έστω ένα instance του VRPTW με:

- Πελάτες $C = \{c_1, ..., c_n\}$ με χρονικά παράθυρα $[e_i, l_i]$
- Όχημα με χωρητικότητα Q
- Depot c_0

Κατασκευάζουμε ένα instance του BDSP ως εξής:

1. Κάθε πελάτης c_i γίνεται δρομολόγιο t_i με $s_{t_i} = e_i, e_{t_i} = l_i$.
2. Το depot γίνεται έδρα λεωφορείου.

3. Οι απαιτήσεις πελατών μεταφράζονται σε χρόνους οδήγησης.

4. Ο περιορισμός χωρητικότητας μεταφράζεται στον περιορισμό D_{max}.

Αν το BDSP έχει εφικτή λύση που καλύπτει όλα τα t_i, τότε το VRPTW έχει επίσης εφικτή λύση. Επομένως, VRPTW $\leq_p$ BDSP. Αφού το VRPTW είναι NP-πλήρες και η αναγωγή είναι πολυωνυμική, το BDSP είναι επίσης NP-πλήρες.

3.8.2. Πρακτικές Συνέπειες και Μέγεθος Προβλήματος

Λόγω της πολυπλοκότητας αυτής, η εύρεση της βέλτιστης λύσης σε περιορισμένο χρόνο είναι συχνά δυσχερής, καθώς ο απαιτούμενος υπολογιστικός χρόνος αυξάνεται εκθετικά με το μέγεθος της εισόδου. Για το λόγο αυτό, η εξαντλητική αναζήτηση θεωρείται υπολογιστικά μη πρακτική για πραγματικές εφαρμογές.

Στη μελέτη περίπτωσης που εξετάζεται, η οποία βασίζεται στα δεδομένα δοκιμής (`data.json`), το πρόβλημα περιλαμβάνει 387 δρομολόγια και 72 λεωφορεία. Αυτό το χαρακτηριστικό μέγεθος καθιστά τον χώρο των δυνατών λύσεων εξαιρετικά μεγάλο, αποκλείοντας κάθε προσπάθεια για εύρεση του καθολικού βέλτιστου μέσω ακριβών (exact) μεθόδων. Συνεπώς, η χρήση στοχαστικών μεταευρετικών μεθόδων τοπικής αναζήτησης καθίσταται αναγκαία, καθώς επιτρέπουν τον εντοπισμό λύσεων υψηλής ποιότητας σε πολύ σύντομο υπολογιστικό χρόνο.

3.9. Σύνοψη

Σε αυτό το κεφάλαιο διατυπώσαμε το πρόβλημα ως μοντέλο ακέραιου γραμμικού προγραμματισμού, ορίζοντας τις μεταβλητές απόφασης x_{ij}, την αντικειμενική συνάρτηση για την ελαχιστοποίηση του σταθμισμένου κόστους και τους σκληρούς περιορισμούς κάλυψης και χρονικής συνοχής. Παράλληλα, εξετάστηκαν οι χαλαροί περιορισμοί που αφορούν την έδρα και τις επιθυμίες των οδηγών. Η επιβεβαίωση της NP-πληρότητας του προβλήματος θεμελιώνει την ανάγκη για τις ευρετικές μεθόδους που αναλύονται στο επόμενο κεφάλαιο.

4. Μεθοδολογία και Αλγόριθμοι

4.1. Γενική Προσέγγιση του Συστήματος

Η προσέγγιση του προβλήματος χρονοπρογραμματισμού λεωφορείων και οδηγών (BDSP) στοχεύει στην εξισορρόπηση της ταχύτητας εκτέλεσης με την ποιότητα της λύσης. Με βάση την NP-πληρότητα του προβλήματος, επιλέχθηκε μια υβριδική μεθοδολογία δύο σταδίων:

1. Στάδιο Κατασκευής (Construction Phase): Χρήση μιας άπληστης κατασκευαστικής ευρετικής για τη δημιουργία μιας αρχικής, εφικτής λύσης.

2. Στάδιο Βελτιστοποίησης (Optimization Phase): Χρήση του αλγορίθμου Προσομοιωμένης Ανόπτησης (Simulated Annealing) για βελτίωση αρχικής λύσης ή χρήση του LAHC ως ανεξάρτητης διαδικασίας τοπικής αναζήτησης (με προαιρετικό previous_solution).

Αυτή η προσέγγιση είναι σχεδιασμένη ώστε να παρέχει μια έγκυρη λύση σε σύντομο χρόνο, επιτρέποντας παράλληλα τη δυνατότητα περαιτέρω βελτίωσης αν διατεθεί περισσότερος υπολογιστικός χρόνος.

4.2. Περιγραφή του Κύριου Επιλύτη

Ο επιλύτης (solver) αποτελεί το κεντρικό συστατικό του συστήματος. Είναι υλοποιημένος σε Python και χρησιμοποιεί τη βιβλιοθήκη Pydantic για την επικύρωση των μοντέλων και τη διαχείριση των δεδομένων.

4.2.1. Αναμενόμενη Είσοδος (Input Specification)

Το σύστημα δέχεται ως είσοδο δεδομένα σε μορφή JSON (`data.json`). Το αρχείο αυτό αποτελεί την ενοποιημένη πηγή δεδομένων και περιλαμβάνει:

- **trips:** λίστα με όλα τα δρομολόγια προς εκτέλεση
- **resources:** λίστα με τους διαθέσιμους οδηγούς και τα λεωφορεία τους
- **stationTransferMatrix:** πίνακα με χρόνους μετάβασης (deadhead) μεταξύ σταθμών

- **constraints:** γενικές παραμέτρους και περιορισμούς (π.χ. μέγιστος χρόνος οδήγησης)

Τα βασικά πεδία εισόδου για κάθε δρομολόγιο είναι:

- **Trip ID:** μοναδικός αναγνωριστικός κωδικός
- **Start/End Time:** ώρες σε μορφή HHMM
- **Origin/Destination:** κωδικοί σταθμών (π.χ. 001 για Αθήνα)
- **Bus Type:** απαιτούμενος τύπος οχήματος

4.2.2. Αναμενόμενη Έξοδος (Output Specification)

Η έξοδος του επιλύτη είναι ένα δομημένο αντικείμενο (JSON) που περιλαμβάνει:

- **Assignments:** λίστα αναθέσεων με τη σειρά εκτέλεσης ανά οδηγό
- **Metrics,** όπως:
  - `Total Cost`: τελική τιμή αντικειμενικής συνάρτησης
  - `Idle Time`: συνολικός νεκρός χρόνος σε λεπτά
  - `Overtime`: συνολικές υπερωρίες
  - `Unassigned Trips`: λίστα δρομολογίων που δεν ανατέθηκαν
- **Validation Status:** επιβεβαίωση ότι η λύση είναι 100% νόμιμη (K1–K15)

4.2.3. Τεχνική Υλοποίηση

Το λογισμικό είναι γραμμένο σε Python και βασίζεται σε τέσσερις κύριους πυλώνες. Ο High-Level Solver συντονίζει τη διαδικασία μέσω της υπηρεσίας `SolverService`, ενώ ο Legality Checker αναλαμβάνει τον αυτόματο έλεγχο τήρησης των κανόνων. Η διαχείριση και επικύρωση των δεδομένων γίνεται μέσω των Pydantic Models. Για τη μοναδική ταυτοποίηση των εγγραφών στη βάση δεδομένων, χρησιμοποιούνται σύνθετα κλειδιά (Composite Keys) (πρωτεύοντα κλειδιά που αποτελούνται από το συνδυασμό δύο ή περισσότερων στηλών, όπως ο συνδυασμός του κωδικού δρομολογίου (`trip_code`) και του αναγνωριστικού του συγκεκριμένου καθημερινού προγράμματος (`schedule_id`)), γεγονός που επιτρέπει τον μοναδικό προσδιορισμό των δρομολογίων στην πάροδο του χρόνου, αποτρέποντας συγκρούσεις κλειδιών

μεταξύ διαφορετικών ημερών.

4.3. Προ-επεξεργασία και Αλυσίδες Δρομολογίων (Chaining)

Μια κρίσιμη φάση πριν την εφαρμογή των αλγορίθμων βελτιστοποίησης είναι η προεπεξεργασία των δεδομένων, με κύριο άξονα τη διαχείριση των υποχρεωτικών συνδέσεων (Joints/Pairs). Στο επιχειρησιακό περιβάλλον των ΚΤΕΛ, ορισμένα δρομολόγια ορίζονται ως αδιαίρετες ακολουθίες που πρέπει να εκτελεστούν από τον ίδιο πόρο διαδοχικά.

Το σύστημα εντοπίζει αυτές τις σχέσεις από τις παραμέτρους `mandatorySequences` και `mandatoryConnections` του αρχείου εισόδου και τις μετατρέπει σε **αλυσίδες (chains)**, δημιουργώντας σύνθετα «μετα-δρομολόγια» (meta-trips). Η διαδικασία αυτή υλοποιείται μέσω της συνάρτησης `_build_chain_mapping` και περιλαμβάνει:

- **Αναδρομική Κατασκευή Αλυσίδων:** Το σύστημα χαρτογραφεί τις σχέσεις εισερχόμενου-εξερχόμενου δρομολογίου και κατασκευάζει μονοπάτια (paths) μοναδικών διαδρομών. Κάθε δρομολόγιο που ανήκει σε μια αλυσίδα αποκτά έναν δείκτη προς το σύνολο των μελών της, διασφαλίζοντας ότι οποιαδήποτε κίνηση (μετακίνηση ή ανταλλαγή) θα αφορά το σύνολο της αλυσίδας.
- **Έλεγχο Εφικτότητας Μεταβάσεων:** Κατά την κατασκευή μιας αλυσίδας, ο αλγόριθμος επαληθεύει ότι ο ενδιάμεσος χρόνος μεταξύ των τμημάτων επαρκεί για την απαραίτητη μετακίνηση του λεωφορείου (deadhead/transfer time), χρησιμοποιώντας τον πίνακα `stationTransferMatrix`. Έτσι, διασφαλίζεται ότι η αλυσίδα είναι εξ ορισμού εκτελέσιμη πριν καν εισαχθεί στον solver.
- **Μείωση Διαστάσεων του Προβλήματος:** Αντιμετωπίζοντας την αλυσίδα ως ενιαία οντότητα με συνδυασμένη διάρκεια και σταθερούς σταθμούς αναχώρησης/άφιξης, μειώνεται δραστικά ο αριθμός των μεταβλητών απόφασης. Για παράδειγμα, μια αλυσίδα τριών δρομολογίων μειώνει τις πιθανές αναθέσεις από $3 \times m$ σε $1 \times m$, βελτιώνοντας σημαντικά την ταχύτητα σύγκλισης των αλγορίθμων.

Η τεχνική αυτή εγγυάται την απόλυτη τήρηση των περιορισμών διαδοχικότητας χωρίς την ανάγκη προσθήκης πολύπλοκων μαθηματικών όρων στην αντικειμενική συνάρτηση, καθιστώντας τον χώρο αναζήτησης πιο «καθαρό» από μη εφικτές

λύσεις.

4.4. Κατασκευαστική Ευρετική (Greedy Heuristic)

Η κατασκευαστική ευρετική στοχεύει στην εύρεση μιας αρχικής λύσης. Η διαδικασία ξεκινά με την ταξινόμηση των δρομολογίων κατά αύξουσα σειρά ώρας έναρξης. Στη συνέχεια, για κάθε δρομολόγιο αναζητείται το πρώτο διαθέσιμο λεωφορείο που ικανοποιεί τους περιορισμούς. Στην υλοποίηση και στα δεδομένα που εξετάστηκαν, ο αλγόριθμος πέτυχε πλήρη κάλυψη των δρομολογίων, παρέχοντας μια βάση για τη βελτιστοποίηση.

4.5. Βελτιστοποίηση με LAHC και SA

Μετά την αρχική λύση, το σύστημα εφαρμόζει δύο αλγορίθμους βελτιστοποίησης. Και στις δύο περιπτώσεις, η αναζήτηση πραγματοποιείται πάνω στην ίδια γειτονιά κινήσεων (relocate, swap) και η διαφορά τους αφορά κυρίως τον κανόνα αποδοχής νέας λύσης.

Στην Προσομοιωμένη Ανόπτηση (Simulated Annealing), η αναζήτηση καθοδηγείται από τη θερμοκρασία T, η οποία επιτρέπει τη στοχαστική αποδοχή χειρότερων λύσεων για την αποφυγή τοπικών ακροτάτων. Για μια υποψήφια κίνηση που επιφέρει μεταβολή κόστους $\Delta = f(s') - f(s)$, ο κανόνας αποδοχής Metropolis ορίζεται ως:

$$P(accept) = \begin{cases} 1, & \text{αν } \Delta \leq 0 \\ e^{-\Delta/T}, & \text{αν } \Delta > 0 \end{cases}$$

Η συγκεκριμένη υλοποίηση ενσωματώνει προηγμένους μηχανισμούς για την ενίσχυση της απόδοσης:

- **Υβριδική Τροχιά Αναζήτησης:** Το σύστημα διατηρεί δύο παράλληλες καταστάσεις, την κύρια (main) και τον βελτιωτή (refiner). Η κύρια τροχιά ακολουθεί το γεωμετρικό πρόγραμμα ψύξης, ενώ ο βελτιωτής λειτουργεί σε σταθερά χαμηλή θερμοκρασία ($T_{ref} = 2.0$), εστιάζοντας στην εντατικοποίηση (intensification) της αναζήτησης σε περιοχές υψηλής ποιότητας.
- **Προσαρμοστικό Πρόγραμμα Ψύξης:** Χρησιμοποιείται γεωμετρική μείωση $T_{next} = T \times \alpha$. Για την επιτάχυνση της σύγκλισης, ο ρυθμός ψύξης

μεταβάλλεται δυναμικά: από $\alpha = 0.9999$ σε υψηλές θερμοκρασίες, σε $\alpha = 0.998$ όταν η θερμοκρασία πέσει κάτω από το κατώφλι των 5.0 μονάδων.

- **Μηχανισμός Επαναθέρμανσης (Reheating):** Σε περίπτωση στασιμότητας (όταν δεν βρεθεί νέα καλύτερη λύση για 5.000 επαναλήψεις), η θερμοκρασία αυξάνεται απότομα στο 50% της αρχικής (T_{start}), επιτρέποντας στον αλγόριθμο να «δραπετεύσει» από βαθιά τοπικά ελάχιστα.
- **Κατευθυνόμενη Επιλογή (Cost-biased Search):** Αντί για πλήρως τυχαία επιλογή οδηγού προς τροποποίηση, ο αλγόριθμος επιλέγει με πιθανότητα 40% από τη λίστα των 10 «χειρότερων» οδηγών (αυτών με το υψηλότερο κόστος βάρδιας). Αυτό κατευθύνει την υπολογιστική προσπάθεια στα σημεία που χρήζουν μεγαλύτερης βελτίωσης.
- **Κινήσεις Φιλικές προς Αλυσίδες (Chain-aware Moves):** Όπως αναλύθηκε στην ενότητα 4.3, όλες οι κινήσεις αντιμετωπίζουν τις αλυσίδες ως αδιαίρετες μονάδες, διασφαλίζοντας τη διατήρηση των υποχρεωτικών συνδέσεων.

Στον αλγόριθμο Late Acceptance Hill Climbing (LAHC), η αποδοχή μιας νέας λύσης s' δεν βασίζεται σε πιθανότητες, αλλά σε μια σύγκριση με το ιστορικό των τιμών κόστους. Ο αλγόριθμος διατηρεί έναν buffer ιστορικού H μήκους L, ο οποίος περιλαμβάνει τις τιμές κόστους των προηγούμενων L αποδεκτών λύσεων.

Ο κανόνας αποδοχής για την επανάληψη it είναι:

$$f(s') \leq H[it \pmod L]$$

Τα βασικά χαρακτηριστικά της υλοποίησης LAHC περιλαμβάνουν:

- **Στρατηγική Προτεραιότητας Κάλυψης (Coverage-first):** Ειδικά για το πρόβλημά μας, οποιαδήποτε κίνηση μειώνει το πλήθος των μη ανατεθειμένων δρομολογίων (`unassigned trips`) γίνεται **πάντα** αποδεκτή, ανεξάρτητα από τον κανόνα του buffer. Αυτό εξασφαλίζει ότι ο αλγόριθμος θα δώσει προτεραιότητα στην εφικτότητα της λύσης.
- **Μηχανισμός Επανεκκίνησης (Restart):** Σε περίπτωση στασιμότητας 20.000

επαναλήψεων, ο αλγόριθμος επαναφέρει την τρέχουσα λύση στην καλύτερη έως τότε γνωστή (best-so-far) και αρχικοποιεί ξανά τον buffer ιστορικού με την τρέχουσα τιμή κόστους.

- **Τερματισμός Δύο Φάσεων:** Η πρώτη φάση στοχεύει στην επίτευξη 100% κάλυψης (μηδενισμός unassigned) εντός του χρονικού ορίου. Μόλις επιτευχθεί αυτός ο στόχος, ο αλγόριθμος εισέρχεται στη δεύτερη φάση, όπου ο buffer καθαρίζεται και η αναζήτηση συνεχίζεται για επιπλέον επαναλήψεις με αποκλειστικό στόχο τη βελτιστοποίηση του λειτουργικού κόστους (νεκρός χρόνος, υπερωρίες).

4.5.1. Σύγκριση Μεθοδολογιών Υλοποίησης

Από την ανάλυση του κώδικα προκύπτει ότι ο LAHC ακολουθεί την κλασική του μορφή, ενώ ο SA ενσωματώνει υβριδικά χαρακτηριστικά. Ο SA λειτουργεί ως ένα πλαίσιο βελτιστοποίησης που συνδυάζει στοχαστική αναζήτηση, ελιτισμό και ευρετικές κατεύθυνσης κόστους. Αντίθετα, ο LAHC βασίζεται στον buffer ιστορικού για την αποδοχή προσωρινών αυξήσεων κόστους, προσφέροντας μια πιο ντετερμινιστική προσέγγιση στη διαφυγή από τοπικά ακρότατα.

4.5.2. Κινήσεις Βελτίωσης

Οι βελτιώσεις επιτυγχάνονται μέσω δύο τύπων κινήσεων:

1. **Μετακίνηση (Relocate):** Ένα δρομολόγιο (ή αλυσίδα) αφαιρείται από το πρόγραμμα ενός οδηγού και τοποθετείται σε έναν άλλο, εφόσον δεν παραβιάζονται οι περιορισμοί.
2. **Ανταλλαγή (Swap):** Δύο οδηγοί ανταλλάσσουν μεταξύ τους δρομολόγια ή αλυσίδες. Η κίνηση αυτή είναι ιδιαίτερα αποτελεσματική για την ανακατανομή του φόρτου εργασίας χωρίς την ανάγκη εύρεσης κενών χρονικών θυρίδων.

Και οι δύο τύποι κινήσεων ελέγχονται αυστηρά από τον Legality Checker πριν την οριστικοποίησή τους.

4.6. Έλεγχος Νομιμότητας (Legality Checker)

Ο μηχανισμός ελέγχου νομιμότητας αποτελεί τον «φύλακα» του συστήματος. Κάθε υποψήφια κίνηση των αλγορίθμων βελτιστοποίησης περνά από έναν εξαντλητικό

έλεγχο 15 κανόνων (K1–K15), οι οποίοι αντιστοιχούν στους περιορισμούς που ορίστηκαν στο Κεφάλαιο 3. Οι κανόνες αυτοί υλοποιούνται στη συνάρτηση `check_assignment_legality` και συνοψίζονται παρακάτω:

- **K1–K3 (Συμβατότητα):** Έλεγχος τύπου λεωφορείου, διαθεσιμότητας οδηγού (Repo/Day-off) και συμβατότητας με την έδρα (Home Station).
- **K4–K5 (Χρονικά Όρια):** Διασφάλιση ότι η βάρδια δεν υπερβαίνει τις 15 ώρες (D_{max}) και ο καθαρός χρόνος εργασίας τις 12 ώρες (W_{max}).
- **K6–K7 (Κανόνες Διαδοχής):** Έλεγχος ειδικών θέσεων βάρδιας (π.χ. δρομολόγια που πρέπει να είναι πρώτα ή τελευταία) και τήρηση της ανάπαυσης μεταξύ διαδοχικών ημερών.
- **K8–K10 (Διαλείμματα και Καθυστερήσεις):** Περιορισμός του αριθμού των μεγάλων κενών στη βάρδια και επιβολή ελάχιστων καθυστερήσεων ανά ομάδα δρομολογίων ή τύπο οχήματος.
- **K11 (Υποχρεωτικό Διάλειμμα):** Επιβολή διαλείμματος τουλάχιστον 30 λεπτών μετά από 4.5 ώρες οδήγησης, σύμφωνα με την ευρωπαϊκή νομοθεσία.
- **K12–K15 (Ειδικοί Περιορισμοί):** Διαχείριση επιθυμιών οδηγών, απαγορευμένων διαδρομών ανά όχημα/οδηγό και προγραμματισμός επιστροφής για συντήρηση (Days to Repo).

Ο Legality Checker λειτουργεί ως σκληρό φίλτρο: αν έστω και ένας από τους 15 κανόνες παραβιαστεί, η κίνηση απορρίπτεται αμέσως, διασφαλίζοντας ότι η αναζήτηση περιορίζεται αποκλειστικά στον χώρο των εφικτών λύσεων.

4.7. Εναλλακτικοί Αλγόριθμοι και Σύγκριση

Παρά την επιλογή των SA και LAHC για το τελικό σύστημα, η βιβλιογραφία προτείνει μια σειρά από εναλλακτικές μεθοδολογίες, οι οποίες εξετάστηκαν κατά τη φάση του σχεδιασμού.

4.7.1. Γενετικοί Αλγόριθμοι (Genetic Algorithms - GA)

Οι Γενετικοί Αλγόριθμοι βασίζονται στις αρχές της φυσικής επιλογής. Αν και είναι ιδιαίτερα αποτελεσματικοί στην εξερεύνηση (exploration) του χώρου λύσεων,

παρουσιάζουν σημαντικές προκλήσεις στο συγκεκριμένο πρόβλημα:

- **Διατήρηση Νομιμότητας:** Η εφαρμογή τελεστών διασταύρωσης (crossover) και μετάλλαξης (mutation) τείνει να καταστρέφει τη χρονική και χωρική συνοχή των βαρδιών, οδηγώντας σε μη εφικτές λύσεις που απαιτούν πολύπλοκους μηχανισμούς διόρθωσης (repair functions).
- **Υπολογιστικό Κόστος:** Η διατήρηση ενός πληθυσμού λύσεων απαιτεί σημαντική μνήμη και επεξεργαστική ισχύ σε σχέση με τις μεθόδους τοπικής αναζήτησης μιας τροχιάς (single-trajectory).

4.7.2. Αναζήτηση Tabu (Tabu Search - TS)

Η αναζήτηση Tabu χρησιμοποιεί βραχυπρόθεσμη μνήμη (Tabu List) για να εμποδίσει την επιστροφή σε πρόσφατα επισκεφθείσες λύσεις, αποφεύγοντας έτσι τους κύκλους. Η μέθοδος αυτή είναι παρόμοια με τον LAHC αλλά πιο «αυστηρή», καθώς απαγορεύει ρητά ορισμένες κινήσεις αντί να βασίζεται σε ιστορικές τιμές κόστους.

4.7.3. Αναζήτηση Μεγάλης Γειτονιάς (Large Neighborhood Search - LNS)

Η LNS και η παραλλαγή της Adaptive LNS (ALNS) βασίζονται στη λογική «καταστροφής και ανακατασκευής» (ruin-and-recreate). Ο αλγόριθμος αφαιρεί τυχαία ή στρατηγικά ένα μεγάλο ποσοστό δρομολογίων από το τρέχον πρόγραμμα και στη συνέχεια προσπαθεί να τα επαναφέρει χρησιμοποιώντας άπληστες ή στοχαστικές ευρετικές. Η μέθοδος αυτή είναι εξαιρετικά ισχυρή για τη διαφυγή από τοπικά ακρότατα αλλά απαιτεί προσεκτική σχεδίαση των τελεστών αφαίρεσης και εισαγωγής.

4.7.4. Αναζήτηση Μεταβλητής Γειτονιάς (Variable Neighborhood Search - VNS)

Η VNS εναλλάσσει συστηματικά τη δομή της γειτονιάς (π.χ. από απλό relocation σε 2-opt ή 3-way exchange) όταν η αναζήτηση σταματά να βελτιώνεται. Η προσέγγισή μας ενσωματώνει στοιχεία αυτής της λογικής μέσω των κινήσεων Relocate και Swap, οι οποίες αποτελούν διαφορετικές γειτονιές αναζήτησης.

Η επιλογή των SA και LAHC έγινε λόγω της ικανότητάς τους να διαχειρίζονται τον αυστηρό έλεγχο νομιμότητας (K1-K15) με ελάχιστη υπολογιστική επιβάρυνση,

προσφέροντας παράλληλα εξαιρετικά αποτελέσματα σε πραγματικά δεδομένα.

4.8. Κριτήρια Τερματισμού

Τα κριτήρια τερματισμού καθορίζουν τη στιγμή κατά την οποία οι αλγόριθμοι σταματούν την αναζήτηση και επιστρέφουν την καλύτερη λύση που έχει βρεθεί. Για τον κατασκευαστικό ευρετικό αλγόριθμο (Greedy), ο τερματισμός είναι άμεσος και συμβαίνει με την ολοκλήρωση της επεξεργασίας όλων των δρομολογίων της εισόδου.

Στον αλγόριθμο Προσομοιωμένης Ανόπτησης (SA), ο τερματισμός επιτυγχάνεται είτε με τη συμπλήρωση του μέγιστου αριθμού επαναλήψεων (`max_iters` = 200.000), είτε όταν η θερμοκρασία πέσει κάτω από το όριο των 0,1 βαθμών. Επιπλέον, εφαρμόζεται μηχανισμός επαναθέρμανσης (reheating) εάν το σύστημα παραμείνει στάσιμο για 5.000 επαναλήψεις.

Για τον αλγόριθμο LAHC, ο τερματισμός καθορίζεται κυρίως από ένα χρονικό όριο 240 δευτερολέπτων. Ωστόσο, εάν επιτευχθεί λύση με πλήρη κάλυψη δρομολογίων (unassigned = 0), ο αλγόριθμος συνεχίζει για ένα επιπλέον πλήθος επαναλήψεων ώστε να βελτιστοποιήσει το κόστος αδράνειας και την επιστροφή στη βάση. Σε περίπτωση στασιμότητας 20.000 επαναλήψεων, πραγματοποιείται επανεκκίνηση (restart) από την καλύτερη αποθηκευμένη λύση.

5. Υλοποίηση Συστήματος

5.1. Αρχιτεκτονική Υψηλού Επιπέδου

Η υλοποίηση βασίζεται σε αρχιτεκτονική τριών επιπέδων: επίπεδο διεπαφής χρήστη (frontend), επίπεδο υπηρεσιών API (backend) και επίπεδο βελτιστοποίησης (solver services). Η διεπαφή χρήστη υποβάλλει αιτήματα επίλυσης μέσω HTTP, ο backend αναλαμβάνει την προεπεξεργασία και την ενοποίηση των δεδομένων, και οι αλγόριθμοι επίλυσης επιστρέφουν αναθέσεις και μετρικές για οπτικοποίηση.

Η προσέγγιση αυτή υποστηρίζει σαφή διαχωρισμό ευθυνών. Το frontend εστιάζει στην αλληλεπίδραση και την παρουσίαση αποτελεσμάτων, ενώ το backend εστιάζει στην επιχειρησιακή λογική και στον έλεγχο περιορισμών.

5.2. Υλοποίηση Backend

5.2.1. REST Endpoints και Ροή Επίλυσης

Ο backend υλοποιείται σε FastAPI και εκθέτει διακριτά endpoints για τις βασικές μεθόδους επίλυσης: `/solve`, `/solve/legacy` και `/solve/lahc`. Κάθε endpoint δέχεται τις αντίστοιχες παραμέτρους εκτέλεσης, εκτελεί τον κατάλληλο αλγόριθμο και επιστρέφει δομημένη απόκριση JSON.

Η τυπική ροή περιλαμβάνει ανάγνωση δεδομένων, κατασκευή αρχικής λύσης (όπου απαιτείται), εφαρμογή βελτιστοποίησης και τελική εξαγωγή μετρικών. Για τις στοχαστικές μεθόδους χρησιμοποιούνται παραμετροποιήσιμα κριτήρια τερματισμού, ώστε η διαδικασία να παραμένει αναπαραγώγιμη στο εκάστοτε πειραματικό πρωτόκολλο.

5.2.2. Υπηρεσίες Επίλυσης και Έλεγχος Νομιμότητας

Η υπηρεσία κατασκευαστικής ευρετικής εκτελεί αρχική ανάθεση δρομολογίων με βάση χρονικούς και χωρικούς περιορισμούς. Η υπηρεσία SA εφαρμόζεται ως φάση βελτιστοποίησης πάνω σε λύση εκκίνησης, ενώ η υπηρεσία LAHC μπορεί να εκτελεστεί είτε από το μηδέν είτε από προϋπάρχουσα λύση μέσω του `previous_solution`.

Ο έλεγχος νομιμότητας εκτελείται σε κάθε κρίσιμο βήμα ανάθεσης. Η λογική αυτή περιορίζει την παραγωγή μη εφικτών λύσεων και επιτρέπει την επιστροφή αποτελεσμάτων που είναι συνεπή με τους κανόνες του προβλήματος.

5.2.3. Δομή Εξόδου και Μετρικές

Η απόκριση του backend περιλαμβάνει τις λίστες αναθέσεων και κατάσταση πόρων, καθώς και ένα σύνολο μετρικών που χρησιμοποιούνται για συγκρίσεις μεταξύ αλγορίθμων. Ενδεικτικά, οι βασικές μετρικές που εξάγονται είναι: πλήθος ανατεθειμένων/μη ανατεθειμένων δρομολογίων, αριθμός χρησιμοποιημένων λεωφορείων, συνολικός νεκρός χρόνος, συνολικός χρόνος βαρδιών, συνολικός χρόνος μετακινήσεων και δείκτης αποκλίσεων βάσης.

5.3. Υλοποίηση Frontend

5.3.1. Οργάνωση Διεπαφής

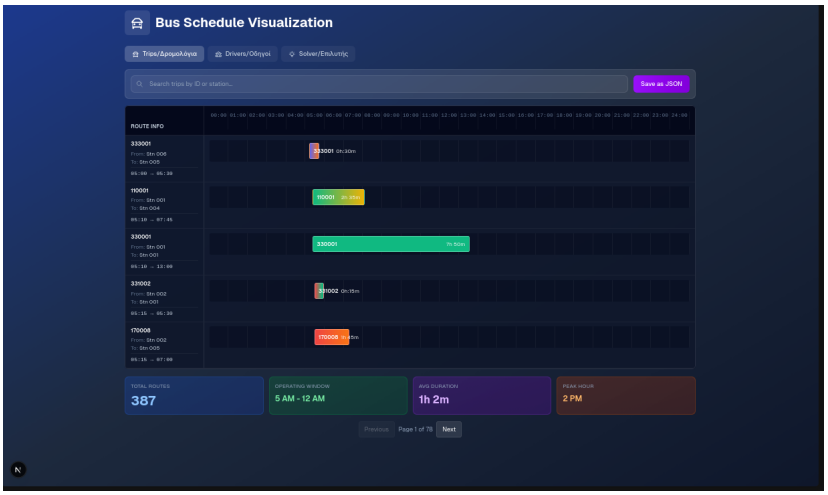
Το frontend υλοποιείται με Next.js και TypeScript. Η κύρια σελίδα οργανώνεται σε καρτέλες για προβολή δρομολογίων, οδηγών και αποτελεσμάτων επίλυσης. Ο χρήστης μπορεί να εκκινήσει διαφορετικούς τρόπους επίλυσης, να συγκρίνει λύσεις και να εξετάσει αναλυτικά χαρακτηριστικά ανά δρομολόγιο ή πόρο.

Η σχεδίαση της διεπαφής στοχεύει σε άμεση εποπτεία των κρίσιμων στοιχείων: χρονικά διαστήματα, αντιστοιχίσεις, και συγκεντρωτικές μετρικές. Με τον τρόπο αυτό, η διαδικασία αξιολόγησης των λύσεων είναι περισσότερο διαφανής.

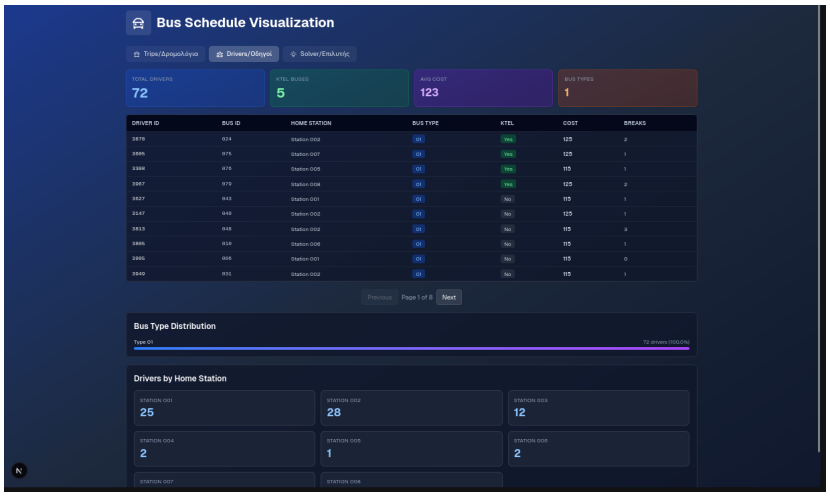
5.3.2. Οπτικοποίηση και Αλληλεπίδραση

Η διεπαφή παρέχει οπτική απεικόνιση τύπου χρονογραμμής για τα δρομολόγια και τις αναθέσεις, καθώς και φίλτρα αναζήτησης ανά σταθμό και αναγνωριστικό δρομολογίου. Επιπλέον, υποστηρίζεται εξαγωγή δεδομένων και συγκριτική προβολή ιστορικού λύσεων για διαφορετικούς αλγορίθμους.

Για την τεκμηρίωση της υλοποίησης παρουσιάζονται στη συνέχεια ενδεικτικά στιγμιότυπα της διεπαφής. Στο Σχήμα 5.1 παρουσιάζονται οι καρτέλες δρομολογίων και οδηγών, ενώ στο Σχήμα 5.2 παρουσιάζεται το πάνελ επίλυσης και η συγκριτική απεικόνιση αποτελεσμάτων των αλγορίθμων.

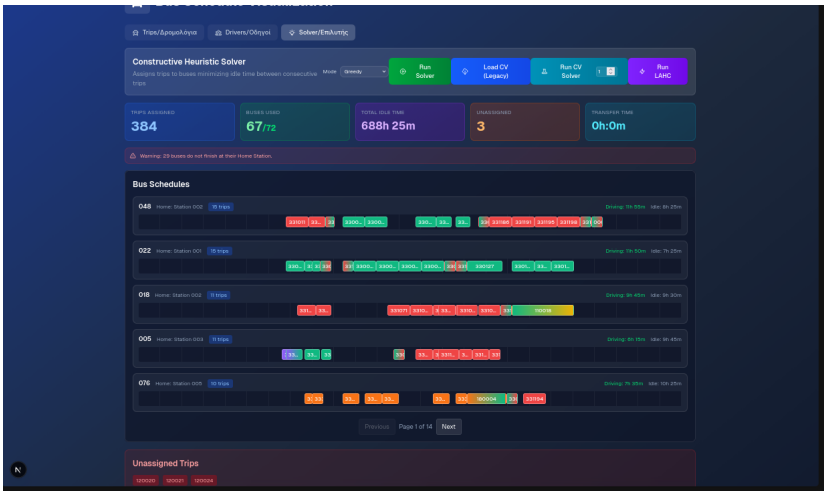


(a) Καρτέλα δρομολογίων.

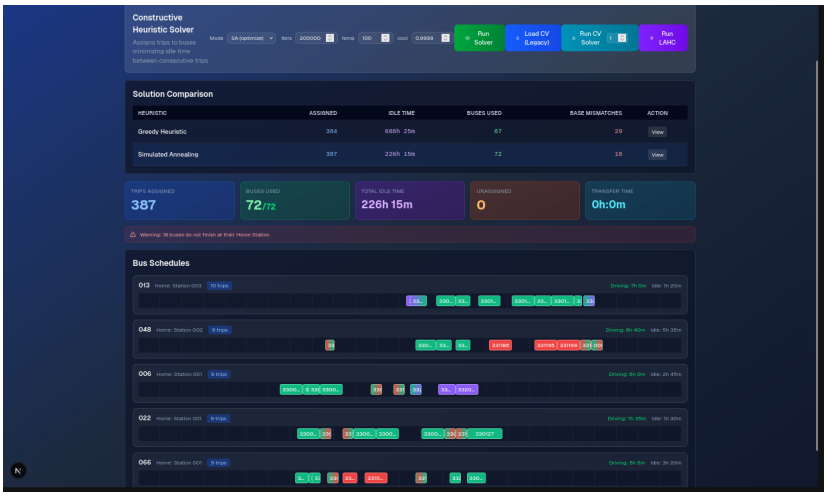


(b) Καρτέλα οδηγών.

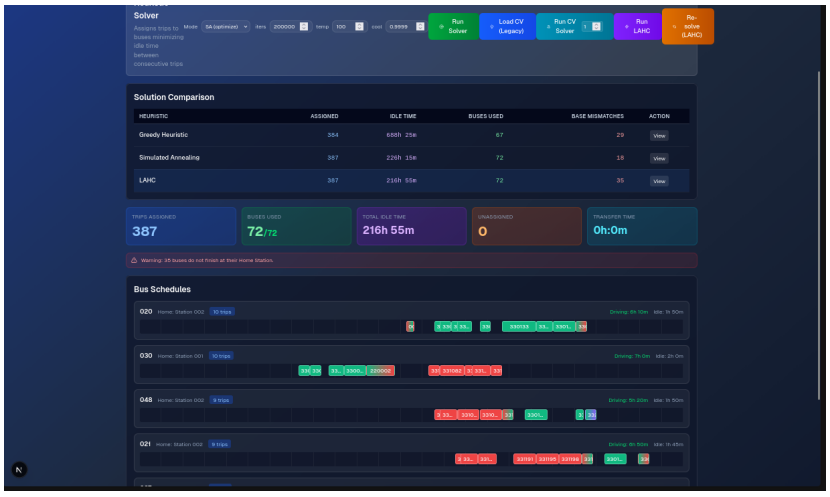
Σχήμα 5.1: Ενδεικτικές οθόνες διεπαφής για εποπτεία δρομολογίων και οδηγών.



(a) Greedy.



(b) SA.



(c) LAHC.

Σχήμα 5.2: Στιγμιότυπα αποτελεσμάτων solver από τη διεπαφή χρήστη.

5.3.3. Διαχείριση και Εξαγωγή Αποτελεσμάτων Επίλυσης (JSON Export)

Μια πρακτική ανάγκη που προέκυψε κατά την ανάπτυξη ήταν η δυνατότητα αποθήκευσης μιας λύσης για μελλοντική χρήση ή σύγκριση. Ένα αποτέλεσμα επίλυσης φαίνεται ενδιαφέρον στη διεπαφή, αλλά αν δεν μπορεί να αποθηκευτεί εκτός βραχύβιας κατάστασης του browser, η πρακτική αξία του περιορίζεται. Για τον λόγο αυτό εντάχθηκε λειτουργία εξαγωγής σε JSON απευθείας από το frontend.

Η εξαγωγή γίνεται από δύο σημεία: στον πίνακα σύγκρισης αλγορίθμων, κάθε γραμμή φέρει ένα κουμπί λήψης που αντιστοιχεί στη συγκεκριμένη εκτέλεση. Εναλλακτικά, στο πάνελ της τρέχουσας ενεργής λύσης, πάνω από τις μετρικές, βρίσκεται ένα δεύτερο κουμπί για άμεση λήψη. Το αποθηκευμένο αρχείο αντικατοπτρίζει ακριβώς τη δομή που επιστρέφει το backend (αναθέσεις, μη ανατεθειμένα δρομολόγια (unassigned), παραμέτρους και μετρικές), κάτι που επιτρέπει την επαναφόρτωση ή επαλήθευση της λύσης με εξωτερικά εργαλεία.

5.4. Σύνοψη

Το κεφάλαιο αυτό παρουσίασε τη συνολική υλοποίηση του συστήματος βελτιστοποίησης δρομολογίων από τα δομικά της θεμέλια έως τη διεπαφή χρήστη. Κεντρικό ρόλο έχει ο διαχωρισμός σε δύο ανεξάρτητα επίπεδα: το backend (FastAPI/Python), που φιλοξενεί τους αλγορίθμους και τον έλεγχο νομιμότητας, και το frontend (Next.js/TypeScript), που αναλαμβάνει την παρουσίαση και την αλληλεπίδραση με τον χρήστη.

Στο backend υλοποιήθηκαν τρεις επιλύτες (ο Greedy ως γρήγορη αρχικοποίηση, ο SA με υβριδικό σχήμα θέρμανσης και ο LAHC με ιστορικό αποδεκτού κόστους), καθώς και ο Legality Checker που επαληθεύει τη συμμόρφωση με τους 15 επιχειρησιακούς περιορισμούς (K1–K15) πριν από κάθε ανάθεση. Η επικοινωνία μεταξύ των δύο επιπέδων γίνεται μέσω REST API με δομημένα αντικείμενα JSON που επικυρώνονται από το Pydantic.

Στο frontend δόθηκε έμφαση στη συγκριτική εποπτεία: ο χρήστης μπορεί να εκτελεί διαφορετικούς αλγορίθμους, να παρακολουθεί το ιστορικό εκτελέσεων και να εξάγει οποιαδήποτε λύση σε αρχείο JSON για αρχειοθέτηση ή offline επαλήθευση. Η λειτουργικότητα αυτή κλείνει τον κύκλο από την εισαγωγή δεδομένων έως την

αξιοποίηση του αποτελέσματος εκτός της εφαρμογής.

Η αρχιτεκτονική που επιλέχθηκε δεν είναι τυχαία: η σαφής οριοθέτηση ευθυνών μεταξύ backend και frontend, σε συνδυασμό με την τυποποίηση της μορφής δεδομένων, καθιστά το σύστημα επεκτάσιμο. Νέοι αλγόριθμοι μπορούν να ενταχθούν χωρίς αλλαγές στη διεπαφή, ενώ η ίδια διεπαφή μπορεί να χρησιμοποιηθεί σε διαφορετικά δίκτυα ΚΤΕΛ με μόνη αλλαγή το αρχείο εισόδου.

6. Πειραματική Αξιολόγηση και Αποτελέσματα

6.1. Πειραματικό Πλαίσιο

Η αξιολόγηση εκτελείται στο ίδιο σύνολο δεδομένων εισόδου, ώστε η σύγκριση μεταξύ αλγορίθμων να είναι συνεπής. Οι μέθοδοι που εξετάζονται είναι οι Greedy, SA και LAHC. Για κάθε μέθοδο καταγράφονται μετρικές ποιότητας λύσης και υπολογιστικού κόστους.

6.2. Μετρικές Αξιολόγησης

Για τη συγκριτική αξιολόγηση των αλγορίθμων χρησιμοποιείται ένα ενιαίο σύνολο μετρικών που αποτυπώνει τόσο την εφικτότητα της λύσης όσο και την επιχειρησιακή της ποιότητα. Οι μετρικές ορίζονται στον Πίνακα 6.1.

Πίνακας 6.1: Μετρικές Αξιολόγησης

Μετρική	Ερμηνεία
Assigned / Unassigned Trips	Βαθμός κάλυψης του ημερήσιου προγράμματος και αριθμός δρομολογίων που παραμένουν χωρίς ανάθεση.
Buses Used	Πλήθος πόρων που ενεργοποιούνται για την εξυπηρέτηση του ίδιου συνόλου δρομολογίων.
Total Idle Time	Συνολικός νεκρός χρόνος μεταξύ διαδοχικών αναθέσεων.
Total Driving Time	Συνολικός καθαρός χρόνος οδήγησης για όλους τους χρησιμοποιημένους πόρους.
Total Transfer Time	Συνολικός χρόνος μετακίνησης μεταξύ σταθμών (όπου εφαρμόζεται).
Base Mismatch Count	Περιπτώσεις στις οποίες ο πόρος δεν καταλήγει στην αναμενόμενη βάση.
Solve Time	Συνολικός χρόνος εκτέλεσης αλγορίθμου για μία επίλυση.

6.3. Πρωτόκολλο Συγκριτικών Δοκιμών

Για τη συγκριτική αξιολόγηση ορίζεται κοινό πρωτόκολλο, ώστε όλες οι μέθοδοι να ελέγχονται υπό τις ίδιες συνθήκες εισόδου. Η λογική του πρωτοκόλλου συνοψίζεται στον Πίνακα 6.2.

Στην τρέχουσα διαμόρφωση της εργασίας, η επιλογή $N = 10$ αποτελεί πρακτικό

Πίνακας 6.2: Πρωτόκολλο Συγκριτικών Δοκιμών

Στάδιο	Περιγραφή
1	Εκτέλεση Greedy ως baseline αναφοράς για εφικτότητα και ταχύτητα.
2	Εκτέλεση SA με σταθερό σύνολο παραμέτρων για N επαναλήψεις.
3	Εκτέλεση LAHC με αντίστοιχο χρονικό όριο για N επαναλήψεις.
4	Ενοποίηση αποτελεσμάτων σε συγκριτικό πίνακα και παραγωγή γραφημάτων.

συμβιβασμό μεταξύ στατιστικής σταθερότητας και συνολικού χρόνου εκτέλεσης.

6.4. Παρουσίαση Αποτελεσμάτων

Η παρουσίαση αποτελεσμάτων οργανώνεται σε συμπληρωματικά αντικείμενα ανάλυσης, ώστε να αποτυπώνονται τόσο οι διαφορές στις μετρικές κόστους όσο και οι διαφορές στον χρόνο εκτέλεσης. Τα αντικείμενα αυτά συνοψίζονται στον Πίνακα 6.3.

Πίνακας 6.3: Αντικείμενα Παρουσίασης Αποτελεσμάτων

Αντικείμενο	Σκοπός
Συγκριτικός πίνακας	Συνολική αποτύπωση επίδοσης ανά μέθοδο στην ίδια πειραματική συνθήκη.
Γράφημα Total Idle Time	Οπτική σύγκριση της αποδοτικότητας ως προς τον νεκρό χρόνο.
Γράφημα Solve Time	Οπτική σύγκριση υπολογιστικού κόστους και χρόνου επίλυσης.

Για τη δημιουργία των ποσοτικών αποτελεσμάτων εκτελέστηκαν μεμονωμένα οι αλγόριθμοι Greedy, SA και LAHC με κοινό σύνολο εισόδου (`data.json`). Οι βασικές ρυθμίσεις εκτέλεσης δίνονται στον Πίνακα 6.4.

Πίνακας 6.4: Ρυθμίσεις Εκτέλεσης Πειραμάτων

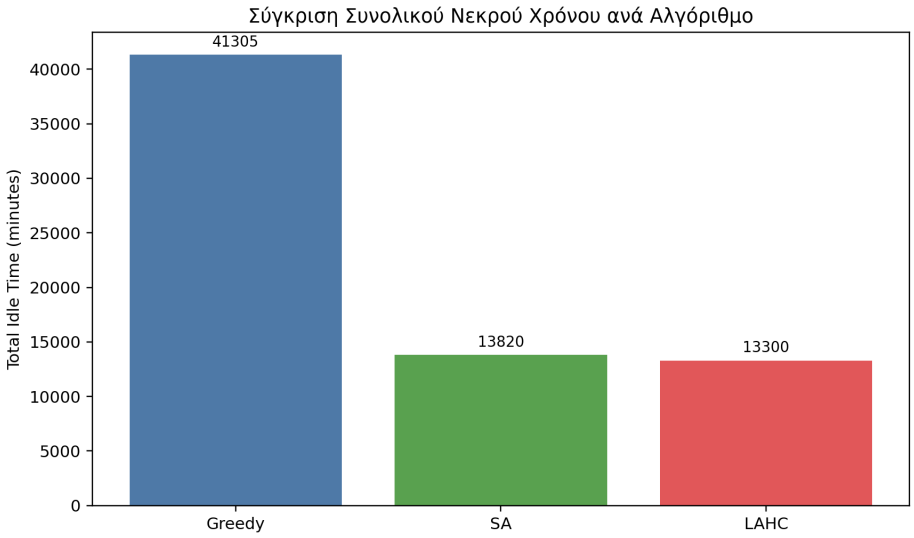
Αλγόριθμος	Ρύθμιση Εκτέλεσης
Greedy	Εκτέλεση μέσω uv run του baseline solver (προεπιλεγμένες ρυθμίσεις).
SA	Εκτέλεση μέσω uv run με max_iters=200000, start_temp=100, cooling=0.9999.
LAHC	Εκτέλεση μέσω uv run με max_iters=300000, L=1500, max_seconds=600.

Τα συγκεντρωτικά αποτελέσματα των τριών εκτελέσεων παρουσιάζονται στον Πίνακα 6.5. Οι τιμές προέρχονται από τα αρχεία εξόδου JSON των εκτελέσεων και αποτελούν το επίσημο benchmark της εργασίας.

Πίνακας 6.5: Συγκριτικά Αποτελέσματα Μεμονωμένων Εκτελέσεων

Αλγόριθμος	Assigned	Unassigned	Buses	Idle	Driving	Mismatch	Wall time (s)
Greedy	384	3	67	41305	23455	29	0.22
SA	387	0	72	13820	23905	16	25.15
LAHC	387	0	72	13300	23905	29	600.00

Η οπτική αποτύπωση των δύο βασικών διαστάσεων σύγκρισης, δηλαδή του συνολικού νεκρού χρόνου και του χρόνου εκτέλεσης, παρουσιάζεται στα Σχήματα 6.1 και 6.2.

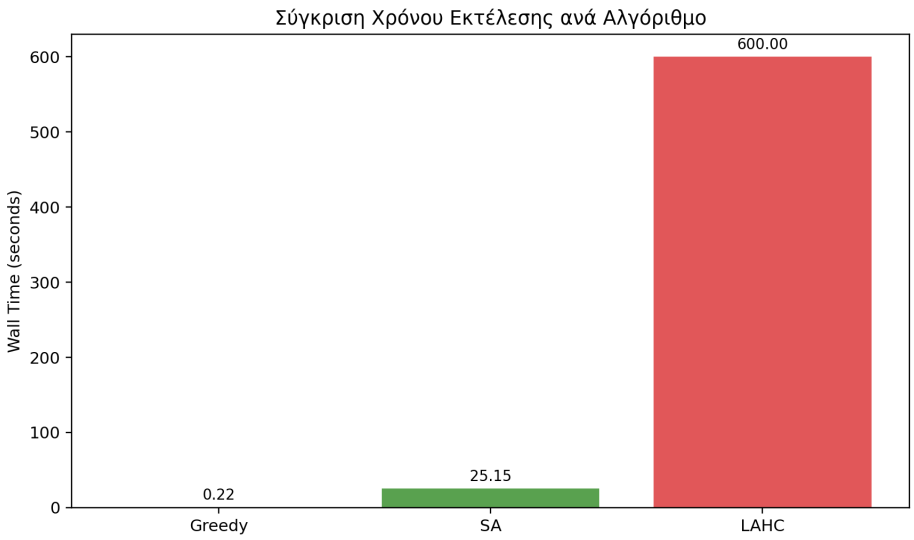


Σχήμα 6.1: Σύγκριση συνολικού νεκρού χρόνου (Total Idle Time) ανά αλγόριθμο.

6.5. Ερμηνεία και Ανάλυση Αποτελεσμάτων

Η ανάλυση των μετρήσεων αποκαλύπτει μια θεμελιώδη διαφορά μεταξύ της κατασκευαστικής ευρετικής και των μεθόδων βελτιστοποίησης. Ενώ ο αλγόριθμος Greedy επιτυγχάνει εξαιρετικά χαμηλούς χρόνους εκτέλεσης (κάτω του δευτερολέπτου), αποτυγχάνει να προσφέρει μια πλήρως εφικτή λύση, αφήνοντας 3 δρομολόγια χωρίς ανάθεση. Στο πλαίσιο της λειτουργίας των ΚΤΕΛ, αυτό το εύρημα είναι κρίσιμο: μια λύση που δεν καλύπτει το σύνολο του ημερήσιου έργου δεν μπορεί να εφαρμοστεί στην πράξη.

Αντίθετα, οι αλγόριθμοι βελτιστοποίησης (SA και LAHC) επιτυγχάνουν πλήρη κάλυψη των δρομολογίων (100%), εξαλείφοντας πλήρως την ποινή ανεκχώρητων δρομολογίων. Η επιτυχία αυτή αναδεικνύει την αξία της μεταευρετικής αναζήτησης,



Σχήμα 6.2: Σύγκριση χρόνου εκτέλεσης (Wall Time) ανά αλγόριθμο.

η οποία είναι σε θέση να αναδιατάσσει τις αναθέσεις με τέτοιο τρόπο ώστε να δημιουργείται «χώρος» για δρομολόγια που αρχικά φαίνονταν αδύνατο να εξυπηρετηθούν λόγω των αυστηρών περιορισμών K1–K15.

Όσον αφορά τις επιμέρους μετρικές:

- **Νεκρός Χρόνος (Idle Time):** Παρατηρείται δραστική μείωση στον νεκρό χρόνο (περίπου 66% βελτίωση σε σχέση με τον Greedy), γεγονός που μεταφράζεται σε πιο συμπαγείς και αποδοτικές βάρδιες.
- **Επιστροφή στη Βάση (Base Mismatch):** Ο αλγόριθμος SA πέτυχε τον μικρότερο αριθμό αποκλίσεων από τη βάση (16 έναντι 29), επιβεβαιώνοντας την αποτελεσματικότητα της ποινής των 2.000 μονάδων που ορίστηκε στην αντικειμενική συνάρτηση.
- **Χρονικό Κόστος:** Παρά την αύξηση στον χρόνο επίλυσης, τα 25 δευτερόλεπτα που απαιτεί ο SA κρίνονται ως απολύτως αποδεκτά για καθημερινή επιχειρησιακή χρήση, προσφέροντας μια βέλτιστη ισορροπία μεταξύ ταχύτητας και ποιότητας αποτελέσματος.

Συμπερασματικά, η μετάβαση από μια απλή άπληστη ανάθεση σε μια υβριδική διαδικασία βελτιστοποίησης δεν αποτελεί απλώς μια ποσοτική αναβάθμιση, αλλά μια ποιοτική μεταβολή που καθιστά το παραγόμενο πρόγραμμα δρομολογίων **λειτουργικά έτοιμο**.

6.5.1. Σε βάθος Σύγκριση: SA vs LAHC

Παρατηρώντας τα συγκεντρωτικά δεδομένα του Πίνακα 6.5, αναδεικνύονται σαφείς διαφορές στη συμπεριφορά των δύο αλγορίθμων, οι οποίες αξίζουν περαιτέρω διερεύνησης. Ο **Simulated Annealing (SA)** κατάφερε να βρει μια εξαιρετική λύση σε μόλις 25 δευτερόλεπτα, ενώ ο **LAHC** χρειάστηκε 600 δευτερόλεπτα για να φτάσει σε ελαφρώς καλύτερο νεκρό χρόνο (13300 έναντι 13820 λεπτών).

Η διαφορά αυτή οφείλεται κυρίως στο υβριδικό σχήμα του SA:

1. **Parallel Trajectories:** Η χρήση του refiner επιτρέπει στον SA να εξερευνά τοπικές βελτιώσεις ενώ η κύρια τροχιά κάνει άλματα σε άλλες περιοχές του χώρου αναζήτησης.
2. **Adaptive Cooling:** Η επιτάχυνση της ψύξης σε χαμηλές θερμοκρασίες επιτρέπει στον SA να κλειδώνει γρήγορα στη βέλτιστη λύση μόλις τη βρει.

Αντίθετα, ο LAHC είναι πιο «υπομονετικός». Λόγω του History Buffer, δεν απορρίπτει εύκολα κινήσεις, κάτι που τον καθιστά ισχυρότερο στην εύρεση του απόλυτου ελαχίστου, αλλά με κόστος σε χρόνο εκτέλεσης. Για τον καθημερινό (offline) προγραμματισμό των δρομολογίων της επόμενης ημέρας, ο LAHC αποτελεί την προτιμότερη επιλογή. Παρόλο που απαιτεί 10 λεπτά εκτέλεσης έναντι των 25 δευτερολέπτων του SA, η εξοικονόμηση 500 λεπτών ($\approx$ 8,3 ωρών) νεκρού χρόνου ανά ημέρα μεταφράζεται σε εξαιρετικά σημαντική μείωση του λειτουργικού κόστους (καύσιμα, φθορές οχημάτων, ανθρωποώρες) σε μεσοπρόθεσμο ορίζοντα, καθιστώντας τον χρόνο αναμονής αμελητέο. Αντίθετα, ο SA κρίνεται ως ο πλέον κατάλληλος για σενάρια δυναμικού επαναπρογραμματισμού σε πραγματικό χρόνο (online rescheduling), όπου απαιτείται άμεση ανταπόκριση (εντός δευτερολέπτων) σε έκτακτα συμβάντα (π.χ. βλάβες οχημάτων ή απουσίες προσωπικού).

6.5.2. Σύγκριση με Άλλη Διπλωματική Εργασία

Για την τοποθέτηση των αποτελεσμάτων σε ευρύτερο πλαίσιο, γίνεται ποιοτική σύγκριση με τη διπλωματική εργασία του Λελούδα [14], η οποία εξετάζει το ίδιο πεδίο με έμφαση στη γρήγορη παραγωγή εφικτής λύσης. Η παρούσα εργασία διαφοροποιείται σε τρία σημεία: (α) ενσωματώνει ρητό έλεγχο νομιμότητας κατά την αναζήτηση, (β) αξιολογεί εναλλακτικές στρατηγικές βελτιστοποίησης στο ίδιο σύνολο

δεδομένων και (γ) παρέχει ενιαία διεπαφή εκτέλεσης και σύγκρισης μετρικών.

Ο Πίνακας 6.6 αποτυπώνει τη σύγκριση σε επίπεδο μεθοδολογίας και πειραματικού πρωτοκόλλου.

Πίνακας 6.6: Ποιοτική Σύγκριση με τη Διπλωματική του Λελούδα

Διάσταση	Λελούδας (2012)	Παρούσα εργασία
Κύρια μεταευρετική προσέγγιση	Γενετικός αλγόριθμος	SA με warm-start από Greedy και LAHC με προαιρετικό previous_solution
Εστίαση αξιολόγησης	Γρήγορη παραγωγή εφικτής λύσης σε προσομοίωση	Ισορροπία εφικτότητας, κόστους και χρόνου εκτέλεσης
Αναφορά μετρικών	Περιγραφή αποτελεσματικότητας σε επίπεδο μελέτης περίπτωσης	Ποσοτικές μετρικές (Assigned/Unassigned, Idle, Wall Time)

Σε σχέση με τα δικά μας αποτελέσματα (Πίνακας 6.5), οι SA και LAHC πέτυχαν πλήρη κάλυψη δρομολογίων (unassigned = 0), με τιμές Total Idle Time 13820 και 13300 αντίστοιχα. Έτσι, η παρούσα εργασία υποστηρίζει άμεση εσωτερική σύγκριση των μεταευρετικών με κοινό πρωτόκολλο και τεκμηριώνει πιο καθαρά τον συμβιβασμό ποιότητας λύσης έναντι χρόνου εκτέλεσης.

Σε μεθοδολογικό επίπεδο, η σύγκριση δείχνει ότι η προσέγγιση της άλλης εργασίας εστιάζει στην παραγωγή μιας εφικτής λύσης σε ένα απλουστευμένο περιβάλλον προσομοίωσης. Παρόλο που οι Γενετικοί Αλγόριθμοι έχουν υψηλή υπολογιστική επιβάρυνση και δεν θεωρούνται «γρήγορες» μέθοδοι λόγω της ανάγκης αξιολόγησης ολόκληρου του πληθυσμού λύσεων, η παράκαμψη των αυστηρών περιορισμών πραγματικού χρόνου διευκολύνει την εύρεση εφικτών λύσεων.

Για πλήρως ποσοτική αντιπαραβολή απαιτείται κοινό πρωτόκολλο αναφοράς (ίδιο σύνολο δεδομένων, ίδια μετρική κόστους και ίδιο υπολογιστικό περιβάλλον), ώστε τα συμπεράσματα να είναι αυστηρά συγκρίσιμα.

7. Περιορισμοί της Μελέτης και Αξιολόγηση Εγκυρότητας

Στο κεφάλαιο αυτό παρουσιάζονται και τεκμηριώνονται οι βασικοί περιορισμοί που επηρεάζουν την αξιοπιστία και τη γενίκευση των συμπερασμάτων της παρούσας εργασίας, ταξινομημένοι ανά κατηγορία εγκυρότητας.

Εσωτερική Εγκυρότητα

Η εσωτερική εγκυρότητα της μελέτης επηρεάζεται από την εξάρτηση των αποτελεσμάτων από ένα συγκεκριμένο σύνολο δεδομένων, το οποίο περιλαμβάνει 387 δρομολόγια και 72 λεωφορεία. Η κατανομή των χρονικών παραθύρων και τα επιχειρησιακά χαρακτηριστικά ενδέχεται να διαφέρουν σε άλλα δίκτυα συγκοινωνιών, ενώ δεν εξετάζονται εποχιακές διακυμάνσεις ή διαφορετικά πρότυπα ζήτησης. Επιπλέον, υιοθετείται η απλοποιητική παραδοχή ότι οι συσχετίσεις οδηγών και λεωφορείων παραμένουν σταθερές κατά τη διάρκεια της ημέρας, παρόλο που στα πραγματικά συστήματα επιτρέπονται εναλλαγές. Τέλος, η στοχαστική φύση των αλγορίθμων Simulated Annealing και Late Acceptance Hill Climbing εισάγει τυχαιότητα λόγω της εξάρτησης από τον αρχικό σπόρο (seed) και της ευαισθησίας στις τιμές των υπερπαραμέτρων (όπως ο ρυθμός ψύξης και το μήκος ιστορικού), χωρίς να έχει πραγματοποιηθεί εξαντλητική αναζήτηση στον χώρο των παραμέτρων αυτών, ενώ ο αριθμός των πειραματικών εκτελέσεων παραμένει περιορισμένος.

Εξωτερική Εγκυρότητα

Όσον αφορά την εξωτερική εγκυρότητα, οι γεωγραφικές και δομικές ιδιαιτερότητες της περιοχής μελέτης περιορίζουν τη δυνατότητα άμεσης γενίκευσης των συμπερασμάτων σε δίκτυα άλλης κλίμακας ή σε συγκοινωνιακά συστήματα διαφορετικών χωρών. Επιπρόσθετα, το υπολογιστικό περιβάλλον εκτέλεσης (εικονική μηχανή Proxmox με τρεις πυρήνες επεξεργαστή Intel Core i5-12500 και μνήμη 6GB RAM) καθορίζει τις μετρήσεις του χρόνου εκτέλεσης, οι οποίες αναμένεται να διαφοροποιηθούν σε εναλλακτικές υποδομές ή σε περιβάλλοντα υπολογιστικού νέφους. Τέλος, ο έλεγχος νομιμότητας δεν ενσωματώνει το σύνολο των ειδικών τοπικών κανονισμών ή των άτυπων πρακτικών που ενδέχεται να εφαρμόζονται σε διαφορετικούς οργανισμούς συγκοινωνιών.

Εγκυρότητα Δομής

Η εγκυρότητα δομής σχετίζεται με τον βαθμό στον οποίο οι επιλεγμένες μετρικές αποτυπώνουν το πραγματικό επιχειρησιακό κόστος. Αν και οι δείκτες του χρόνου αδράνειας, του συνολικού χρόνου εκτέλεσης και των μη καλυπτόμενων δρομολογίων αποτελούν αξιόπιστες προσεγγίσεις της αποδοτικότητας, δεν ενσωματώνουν πλήρως άλλες κρίσιμες παραμέτρους, όπως την κατανάλωση καυσίμων, τη φθορά των οχημάτων και τις υποκειμενικές προτιμήσεις των οδηγών κατά την κατανομή των βαρδιών.

8. Συμπεράσματα και Μελλοντική Εργασία

8.1. Συμπεράσματα

Η παρούσα εργασία αντιμετώπισε με επιτυχία το σύνθετο πρόβλημα του ταυτόχρονου χρονοπρογραμματισμού οχημάτων και οδηγών (Integrated Vehicle and Crew Scheduling Problem) στο πλαίσιο των ιδιωτικών υπεραστικών συγκοινωνιών στην Ελλάδα (ΚΤΕΛ). Όπως αναλύθηκε στο Κεφάλαιο 1 και το Κεφάλαιο 2, ο χρονοπρογραμματισμός στις ελληνικές συγκοινωνίες παρουσιάζει μια σημαντική ιδιαιτερότητα: την ενιαία αντιμετώπιση του οδηγού και του οχήματος ως ενιαίας λειτουργικής μονάδας, κυρίως λόγω του ιδιοκτησιακού καθεστώτος των λεωφορείων. Αυτό διαφοροποιεί το πρόβλημα από τη διεθνή βιβλιογραφία, όπου ο προγραμματισμός οχημάτων και η ανάθεση βαρδιών επιλύονται ως δύο διακριτά και διαδοχικά στάδια. Η ενιαία αυτή διαχείριση μετατρέπει το πρόβλημα σε μια σύνθετη παραλλαγή του προβλήματος δρομολόγησης οχημάτων με χρονικά παράθυρα (VRPTW), όπου οι περιορισμοί χρονικής και χωρικής συνοχής καθίστανται καθοριστικοί για τη βιωσιμότητα των λύσεων.

Στο Κεφάλαιο 3 παρουσιάστηκε η μαθηματική διατύπωση του προβλήματος, η οποία περιλαμβάνει τη μοντελοποίηση των επιχειρησιακών δεδομένων και τη λεπτομερή αποτύπωση των εργασιακών και νομικών περιορισμών. Στο πλαίσιο αυτό, αναπτύχθηκε και ενσωματώθηκε ένας αυστηρός ελεγκτής νομιμότητας (rules checker) αποτελούμενος από δεκαπέντε διακριτούς κανόνες. Οι κανόνες αυτοί καλύπτουν τόσο το εθνικό εργατικό δίκαιο όσο και τις ειδικές συμβατικές υποχρεώσεις των συγκοινωνιακών οργανισμών, ρυθμίζοντας κρίσιμες παραμέτρους όπως τα όρια της ημερήσιας εργασίας, τον μέγιστο συνεχή χρόνο οδήγησης, τη διάρκεια και τη συχνότητα των διαλειμμάτων ανάπαυσης, καθώς και τον υπολογισμό των νυχτερινών βαρδιών και των υπερωριών. Η μοντελοποίηση αυτή επέτρεψε τον ακριβή προσδιορισμό του λειτουργικού κόστους κάθε βάρδιας μέσω μιας σύνθετης αντικειμενικής συνάρτησης, η οποία στοχεύει στην ταυτόχρονη ελαχιστοποίηση των απαιτούμενων οχημάτων, των άσκοπων χιλιομέτρων αδράνειας (deadheads) και των αλλαγών οχημάτων από τους οδηγούς.

Για την επίλυση του προβλήματος, όπως αναλύθηκε στο Κεφάλαιο 4, σχεδιάστηκαν

και υλοποιήθηκαν δύο στοχαστικοί μεταευρετικοί αλγόριθμοι τοπικής αναζήτησης: η Προσομοιωμένη Ανόπτηση (Simulated Annealing - SA) και ο αλγόριθμος Late Acceptance Hill Climbing (LAHC). Οι μεταευρετικές αυτές τεχνικές επιλέχθηκαν λόγω της ικανότητάς τους να διαχειρίζονται τον τεράστιο χώρο αναζήτησης του προβλήματος και να ξεφεύγουν από τοπικά ελάχιστα. Για την αρχικοποίηση των αλγορίθμων αναπτύχθηκε μια κατασκευαστική ευρετική μέθοδος (Greedy heuristic), η οποία παράγει γρήγορα μια πρώτη εφικτή λύση, η οποία στη συνέχεια βελτιστοποιείται από τους μεταευρετικούς αλγορίθμους μέσω δομημένων κινήσεων γειτονιάς (neighborhood moves), όπως η ανταλλαγή δρομολογίων μεταξύ βαρδιών ή η εισαγωγή δρομολογίων σε κενά διαστήματα.

Στο Κεφάλαιο 5 περιγράφηκε η αρχιτεκτονική και η υλοποίηση του ολοκληρωμένου λογισμικού συστήματος. Το σύστημα αναπτύχθηκε με βάση μια σύγχρονη και σπονδυλωτή αρχιτεκτονική, χρησιμοποιώντας Python και FastAPI για το backend και Next.js με TypeScript για το frontend. Η επικοινωνία μεταξύ των υποσυστημάτων πραγματοποιείται μέσω τυποποιημένων αρχείων JSON, εξασφαλίζοντας τη συμβατότητα και τη δυνατότητα μελλοντικής διασύνδεσης με εξωτερικά συστήματα διαχείρισης πόρων (ERP). Επιπλέον, σχεδιάστηκε μια εύχρηστη διεπαφή χρήστη που επιτρέπει στον συγκοινωνιακό φορέα να εισάγει τα δεδομένα των δρομολογίων, να παρακολουθεί την εξέλιξη της βελτιστοποίησης σε πραγματικό χρόνο και να οπτικοποιεί τα παραγόμενα προγράμματα εργασίας και τις διαδρομές των οχημάτων.

Η πειραματική αξιολόγηση του συστήματος, η οποία παρουσιάστηκε στο Κεφάλαιο 6, πραγματοποιήθηκε με τη χρήση πραγματικών δεδομένων από ελληνικούς συγκοινωνιακούς φορείς. Τα αποτελέσματα κατέδειξαν ότι και οι δύο μεταευρετικοί αλγόριθμοι επιτυγχάνουν πλήρη κάλυψη των δρομολογίων, προσφέροντας σημαντική βελτίωση στην ποιότητα των λύσεων σε σχέση με την κατασκευαστική Greedy μέθοδο. Ειδικότερα, ο αλγόριθμος Simulated Annealing επέδειξε εξαιρετική σταθερότητα στην εύρεση λύσεων υψηλής ποιότητας με ελάχιστο χρόνο αδράνειας, ενώ ο LAHC παρουσίασε ταχύτερη σύγκλιση, καθιστώντας τον κατάλληλο για σενάρια όπου απαιτείται άμεση λήψη αποφάσεων. Η σύγκριση των μεθόδων ανέδειξε έναν σαφή συμβιβασμό (trade-off) μεταξύ του απαιτούμενου υπολογιστικού χρόνου και της ποιότητας της παραγόμενης λύσης, προσφέροντας στον χρήστη τη δυνατότητα να επιλέγει την κατάλληλη μέθοδο ανάλογα με τις επιχειρησιακές

ανάγκες.

Συνολικά, η εργασία αυτή προσφέρει μια ολοκληρωμένη και πρακτικά εφαρμόσιμη λύση σε ένα παραδοσιακά δύσκολο πρόβλημα, γεφυρώνοντας το χάσμα μεταξύ της θεωρητικής βελτιστοποίησης και της καθημερινής λειτουργίας των συγκοινωνιακών φορέων στην Ελλάδα. Η δυνατότητα άμεσης και ισότιμης σύγκρισης διαφορετικών αλγορίθμων υπό κοινές συνθήκες αξιολόγησης και μέσα από ένα ενιαίο περιβάλλον αποτελεί τη βασική συνεισφορά του συστήματος, θέτοντας τις βάσεις για τον εκσυγχρονισμό των διαδικασιών προγραμματισμού στις ελληνικές υπεραστικές συγκοινωνίες.

8.2. Μελλοντική Εργασία

Η επέκταση και η βελτίωση του προτεινόμενου συστήματος μπορούν να κινηθούν σε διάφορες ερευνητικές και πρακτικές κατευθύνσεις. Αρχικά, κρίνεται σκόπιμη η εκτενής διερεύνηση του χώρου των υπερπαραμέτρων για τους μεταευρετικούς αλγορίθμους, καθώς και η δοκιμή του συστήματος με πολλαπλά και διαφορετικά σύνολα δεδομένων από άλλους συγκοινωνιακούς φορείς για την επιβεβαίωση της γενικευσιμότητας του. Επιπλέον, η αυτοματοποίηση της παραγωγής συγκριτικών αναφορών και γραφημάτων σύγκλισης θα διευκολύνει την ανάλυση της συμπεριφοράς των αλγορίθμων. Σε επίπεδο αλγοριθμικής σχεδίασης, ο συνδυασμός της Greedy αρχικοποίησης με μεταευρετικές μεθόδους, καθώς και η υποστήριξη δυναμικής ανακατανομής δρομολογίων σε πραγματικό χρόνο κατά τη διάρκεια της ημέρας, αποτελούν σημαντικές κατευθύνσεις εξέλιξης. Τέλος, η διασύνδεση του συστήματος με εξωτερικά ERP ή πλατφόρμες διαχείρισης προσωπικού και συντήρησης οχημάτων, μέσω της τυποποιημένης εξόδου JSON, μπορεί να αυτοματοποιήσει πλήρως τις επιχειρησιακές διαδικασίες.

Παράλληλα, η σταδιακή μετάβαση των συγκοινωνιακών φορέων στην ηλεκτροκίνηση δημιουργεί την ανάγκη για προσαρμογή του μοντέλου στις ιδιαιτερότητες των ηλεκτρικών λεωφορείων. Η ενσωμάτωση αυτή εισάγει πρόσθετους περιορισμούς, όπως η περιορισμένη αυτονομία των οχημάτων, η διακύμανση της κατανάλωσης ενέργειας ανάλογα με τη διαδρομή και τις συνθήκες κυκλοφορίας, καθώς και ο προγραμματισμός των αναγκαίων περιόδων φόρτισης. Η επέκταση του αλγορίθμου για τη βέλτιστη κατανομή των οχημάτων σε σταθμούς φόρτισης,

λαμβάνοντας υπόψη τη διαθεσιμότητα των φορτιστών και το μεταβαλλόμενο κόστος του ηλεκτρικού ρεύματος κατά τη διάρκεια της ημέρας, αποτελεί μια εξαιρετικά ενδιαφέρουσα ερευνητική πρόκληση με άμεσο πρακτικό αντίκτυπο.

Η παρούσα εργασία θέτει ένα πρακτικό και τεχνικά τεκμηριωμένο θεμέλιο για περαιτέρω βελτιώσεις, τόσο σε αλγοριθμικό επίπεδο όσο και σε επίπεδο ενσωμάτωσης σε πραγματικό περιβάλλον λειτουργίας. Η συνδυαστική προσέγγιση των μεταευρετικών αλγορίθμων και του αυστηρού ελέγχου νομιμότητας προσφέρει μια στέρεη βάση για την περαιτέρω ερευνητική εξέλιξη του πεδίου, υποστηρίζοντας ταυτόχρονα την άμεση κάλυψη πραγματικών επιχειρησιακών αναγκών.

Α. Παράρτημα Α: Δείγμα Δεδομένων και Κώδικας

Α.1. Δομή Αρχείου Εισόδου JSON

Το αρχείο εισόδου οργανώνεται στις ενότητες metaData, configuration, stationTransferMatrix, resources και trips. Στο Απόσπασμα Α.1 παρουσιάζεται η κορυφή του αρχείου με τις βασικές παραμέτρους του solver.

Listing A.1: Ενδεικτική κορυφή αρχείου εισόδου JSON.

```
{
  "metaData": {
    "dayOfPeriod": 61,
    "periodLength": 90,
    "description": "Bus scheduling data parsed from DAT files"
  },
  "configuration": {
    "parameters": {
      "maxDutyMinutes": 900,
      "normalDutyMinutes": 600,
      "maxWorkMinutes": 720,
      "driveTimeToBreakMinutes": 270,
      "minBreakMinutes": 30,
      "transitMinutes": 30
    }
  }
}
```

Α.2. Δείγμα Εγγραφών Πόρων και Δρομολογίων

Στο Απόσπασμα Α.2 παρουσιάζεται μία εγγραφή πόρου (συσχέτιση οδηγού/λεωφορείου) και μία εγγραφή δρομολογίου. Τα πεδία αυτά χρησιμοποιούνται άμεσα στους ελέγχους εφικτότητας και στη συνάρτηση κόστους.

Listing A.2: Ενδεικτικές εγγραφές resource και trip.

```
{
  "resource": {
    "driverId": "3878",
    "busId": "024",
    "homeStation": "002",
    "currentStation": "002",
    "busAttributes": { "type": "01", "isKtel": 1 },
    "parameters": { "cost": 125, "compatibility": "+000000" }
  },
  "trip": {
    "id": "333001",
    "startStation": "006",
    "endStation": "005",
    "start_time": "05:00",
    "end_time": "05:30",
    "start_minutes": 300,
    "end_minutes": 330,
    "classification": { "group": "000", "allowedBusTypes": [ "01" ] }
  }
}
```

A.3. Δομή Αρχείου Εξόδου JSON (Λύση)

Αντίθετα με το αρχείο εισόδου, κάθε εκτέλεση αλγορίθμου παράγει ένα αρχείο JSON που περιγράφει πλήρως τη λύση. Η δομή χωρίζεται σε τέσσερα κύρια τμήματα: τις αναθέσεις (`assignments`), τα δρομολόγια που δεν καλύφθηκαν (`unassignedTrips`), τις παραμέτρους επίλυσης (`parameters`) και τις συγκεντρωτικές μετρικές (`metrics`). Κάθε ανάθεση (`assignment`) συνδέει άμεσα ένα δρομολόγιο (`tripId`) με έναν συγκεκριμένο οδηγό (`driverId`) και ένα λεωφορείο (`busId`). Στο Απόσπασμα A.3 φαίνεται ένα χαρακτηριστικό δείγμα αυτής της δομής, βασισμένο σε πραγματική εκτέλεση των αλγορίθμων βελτιστοποίησης.

Listing A.3: Ενδεικτική δομή αρχείου εξόδου JSON (Λύση).


```
{
  "assignments": [
    {
      "tripId": "333001",
      "driverId": "3897",
      "busId": "004"
    }
  ],
  "unassignedTrips": [
    "120020"
  ],
  "parameters": {
    "maxDutyMinutes": 900,
    "normalDutyMinutes": 600,
    "maxWorkMinutes": 720,
    "driveTimeToBreakMinutes": 270,
    "minBreakMinutes": 30,
    "transitMinutes": 30,
    "maxDutyGaps": 5,
    "minGapMinutes": 15
  },
  "metrics": {
    "totalTrips": 387,
    "assigned": 384,
    "unassigned": 3,
    "busesUsed": 67,
    "totalIdleTime": 41305,
    "totalDrivingTime": 23455,
    "totalTransferTime": 0,
    "baseMismatchCount": 29,
    "chainsProcessed": 62,
    "connectionsInData": 114
  }
}
```

A.4. Απόσπασμα Κώδικα Legality Checker

Ο έλεγχος νομιμότητας υλοποιείται στο αρχείο `backend/services/legality_checker.py`. Το Απόσπασμα A.4 δείχνει τον πυρήνα της συνάρτησης `check_assignment_legality`, όπου ορίζονται οι έλεγχοι K1–K15.

Listing A.4: Πυρήνας συνάρτησης check_assignment_legality.

```
def check_assignment_legality(
    trip: Dict,
    resource: Optional[Dict],
    current_trips: List[Dict],
    params: SolverParameters = None,
    day_index: int = 0,
    constraints_data: ConstraintsData = None,
) -> Tuple[bool, List[str]]:
    """
    Implements legacy LC checks:
    - K1: Bus type check
    - K2: Repo/Day-off compatibility
    - K3: Base compatibility check
    - ...
    - K15: Days to Repo check
    """
    if params is None:
        params = SolverParameters()
    if constraints_data is None:
        constraints_data = ConstraintsData()
    # ... logic for checks ...
    return True, []
```

Επιπλέον, η συνάρτηση `get_transfer_duration` αξιοποιεί τον πίνακα μεταβάσεων σταθμών (`stationTransferMatrix`) και επιβάλλει είτε απαιτούμενο διάλειμμα είτε μηδενικό χρόνο μεταφοράς μεταξύ διαδοχικών δρομολογίων.

Α.5. Διαμόρφωση Docker και Εκτέλεση Συστήματος

Η ανάπτυξη και η εκτέλεση του συστήματος βασίζονται στη χρήση κοντέινερ μέσω της πλατφόρμας Docker. Η ενορχήστρωση των επιμέρους υπηρεσιών καθορίζεται στο αρχείο `docker-compose.yml`, το οποίο παρουσιάζεται στο Απόσπασμα Α.5. Το αρχείο αυτό οργανώνει την εφαρμογή σε τρία διακριτά κοντέινερ (frontend, backend και db), τα οποία επικοινωνούν μεταξύ τους μέσω ενός κοινού εσωτερικού δικτύου γέφυρας με την ονομασία `bus_network`.

Η υπηρεσία `frontend` βασίζεται στο Next.js και εκτίθεται στη θύρα 3000, ενώ η υπηρεσία `backend` εκτελεί την εφαρμογή FastAPI στη θύρα 8000 χρησιμοποιώντας τον διακομιστή uvicorn. Η βάση δεδομένων `db` χρησιμοποιεί την επίσημη εικόνα PostgreSQL 16-alpine. Η επικοινωνία μεταξύ των υπηρεσιών είναι άμεση, ενώ η χρήση κοινόχρηστων τόμων (volumes) επιτρέπει την άμεση ενημέρωση του κώδικα κατά τη διάρκεια της ανάπτυξης (live reloading) χωρίς να απαιτείται εκ νέου χτίσιμο των κοντέινερ. Η εκτέλεση ολόκληρου του συστήματος πραγματοποιείται με μια απλή εντολή `docker compose up --build`, απλοποιώντας σημαντικά τη διαδικασία εγκατάστασης σε οποιοδήποτε περιβάλλον.

Listing A.5: Αρχείο ενορχήστρωσης docker-compose.yml.

```
services:
  frontend:
    build: ./frontend
    ports:
      - "3000:3000"
    volumes:
      - ./frontend:/app:Z
      - /app/node_modules
    environment:
      - NODE_ENV=development
    user: "1000:1000"
    networks:
      - bus_network
    depends_on:
      - backend

  backend:
    build: ./backend
    ports:
      - "8000:8000"
    volumes:
      - ./backend:/app:Z
      - ./docs:/app/docs:Z
      - ./data.json:/app/data.json:Z
    env_file:
      - ./backend/.env
    environment:
      - PYTHONPATH=/app
      - UV_CACHE_DIR=/tmp/uv-cache
    command: ["python", "-m", "uvicorn", "main:app", "--host", "0.0.0.0", "--
        ↪ port", "8000", "--reload"]
    networks:
      - bus_network
    depends_on:
      - db
```

Βιβλιογραφία

[1] Christos Valouxis και Efthymios Housos. “Combined bus and driver scheduling”. Στο: Computers & Operations Research 29.3 (2002), σσ. 243–259.

[2] Guy Desaulniers, Jacques Desrosiers, Irina Ioachim, Marius M Solomon, François Soumis και Daniel Villeneuve. “Crew scheduling in air transit”. Στο: Fleet management and scheduling (1998), σσ. 169–185.

[3] Christos Gogos, Christos Valouxis, Panayiotis Alefragis, Georgios Goulas, Nikolaos Voros και Efthymios Housos. “Scheduling independent tasks on heterogeneous processors using heuristics and column pricing”. Στο: Future Generation Computer Systems 60 (2016), σσ. 48–66. DOI: 10.1016/j.future.2016.01.016.

[4] Christos Gogos. “Solving the Distributed Permutation Flow-Shop Scheduling Problem Using Constrained Programming”. Στο: Applied Sciences 13.23 (2023), σ. 12562. DOI: 10.3390/app132312562.

[5] Christos Gogos, Panayiotis Alefragis και Efthymios Housos. “An improved multi-staged algorithmic process for the solution of the examination timetabling problem”. Στο: Annals of Operations Research 194.1 (2012), σσ. 203–221. DOI: 10.1007/s10479-010-0712-3.

[6] Gerhard Post, Jeffrey H Kingston, Samad Ahmadi, Sophia Daskalaki, Christos Gogos, Jari Kyngas, Kimmo Nurmi, Nysret Musliu, Nelishia Pillay, Haroldo Santos και Andrea Schaerf. “XHSTT: an XML archive for high school timetabling problems in different countries”. Στο: Annals of Operations Research 218.1 (2014), σσ. 295–301. DOI: 10.1007/s10479-011-1012-2.

[7] Christos Valouxis, Christos Gogos, Georgios Goulas, Panayiotis Alefragis και Efthymios Housos. “A systematic two phase approach for the nurse rostering problem”. Στο: European Journal of Operational Research 219.2 (2012), σσ. 425–433. DOI: 10.1016/j.ejor.2011.12.042.

[8] George B Dantzig και John H Ramser. “The truck dispatching problem”. Στο: Management science 6.1 (1959), σσ. 80–91.

[9] Cynthia Barnhart, Amy M Cohn, Ellis L Johnson, Diego Klabjan, George L Nemhauser και Pamela H Vance. “Crew scheduling”. Στο: Handbook of transportation science (2003), σσ. 493–521.

[10] Michael R Garey και David S Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman και Company, 1979.

[11] Jan Karel Lenstra και Alexander HG Rinnooy Kan. “Complexity of vehicle routing and scheduling problems”. Στο: Networks 11.2 (1981), σσ. 221–227.

[12] Ademir Aparecido Constantino, Candido Ferreira Xavier de Mendonça Neto, Silvio Alexandre de Araujo, Dario Landa-Silva, Rogério Calvi και Allainclair Flausino dos Santos. “Solving a Large Real-world Bus Driver Scheduling Problem with a Multi-assignment based Heuristic Algorithm”. Στο: Journal of Universal Computer Science 23.5 (2017), σσ. 479–504.

[13] Martin WP Savelsbergh και Marc Sol. “The general pickup and delivery problem”. Στο: Transportation science 29.1 (1995), σσ. 17–29.

[14] Panagiotis Leloudas. “Σχεδιασμός, Ανάλυση και Υλοποίηση Ευφυών Αλγορίθμων Υπολογιστικής Νοημοσύνης για την Εύρεση Βέλτιστου Ωρολογίου Προγράμματος Εργασίας Οδηγών και Χρονοδρομολόγησης Λεωφορείων σε Υπεραστικά και Αστικά ΚΤΕΛ στην Ελλάδα.” Master’s Thesis. University of Patras, 2012. URL: http://hdl.handle.net/10889/5329.